(2)

RL-TR-92-345, Vol II (of seven)
Final Technical Report
December 1992

# SYSTEM ENGINEERING CONCEPT DEMONSTRATION, Systems Engineering Needs

Software Productivity Solutions, Inc.

Edward R. Comer

**Rome Laboratory
Air Force Materiel Command
Griffiss Air Force Base, New York**

RL-TR-92-345, Volume II (of seven) has been reviewed and is approved for publication.

APPROVED: [illegible]

FRANK S. LAMONICA
Project Engineer

FOR THE COMMANDER: [illegible]

JOHN A. GRANIERO
Chief Scientist
Command, Control & Communications Directorate

# REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

| | | |
|---|---|---|
| 1. AGENCY USE ONLY (Leave Blank) | 2. REPORT DATE<br>December 1992 | 3. REPORT TYPE AND DATES COVERED<br>Final Feb 90 - Jul 92 |

**4. TITLE AND SUBTITLE**
SYSTEM ENGINEERING CONCEPT DEMONSTRATION,
Systems Engineering Needs

**5. FUNDING NUMBERS**
C - F30602-90-C-0021
PE - 62702F
PR - 5581
TA - 18
WU - 54

**6. AUTHOR(S)**
Edward R. Comer

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Software Productivity Solutions, Inc.
122 4th Avenue
Indialantic FL 32903-1697

**8. PERFORMING ORGANIZATION REPORT NUMBER**
N/A

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
Rome Laboratory (C3CB)
525 Brooks Road
Griffiss AFB NY 13441-4505

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
RL-TR-92-345, Vol II
(of seven)

**11. SUPPLEMENTARY NOTES**
Rome Laboratory Project Engineer: Frank S. LaMonica/(315) 330-2054

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**
Approved for public release; distribution unlimited.

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** (Maximum 200 words)

This final technical report documents the objectives, activities, and results of Air Force contract F30602-90-C-0021, entitled "System Engineering Concept Demonstration." The effort, which was conducted by Software Productivity Solutions, Inc., with McDonnell Douglas Corporation - Douglas Aircraft Company and MTM Engineering Inc. as subcontractors, demonstrated and documented the concept of an advanced computer-based environment which provides automation for Systems Engineering tasks and activities within the Air Force computer-based systems life cycle. The report consists of seven (7) volumes as follows: I) Effort Summary, II) Systems Engineering Needs, III) Process Model, IV) Interface Standards Studies, V) Technology Assessments, VI) Trade Studies, and VII) Security Study.

This Volume (Volume II - Systems Engineering Needs), describes the various investigations that were conducted to analyze and prioritize the systems engineering needs that the envisioned systems engineering automation will satisfy.

**14. SUBJECT TERMS**
System Engineering, System Life Cycle Tools, System Life Cycle Environment

**15. NUMBER OF PAGES**
140

**16. PRICE CODE**

| | | | |
|---|---|---|---|
| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | UL |

# Table of Contents

## List of Tables

# 1. Introduction

This is a supporting document to the Final Technical Report of the Systems Engineering Concept Demonstration, contract F30602-90-R-0003 for the Air Force Rome Laboratory (RL). The document is organized as follows:

1. Introduction
2. Needs Survey
3. Field Interviews
4. Conclusions

The needs survey investigated the problems with mission-critical systems development and maintenance, and the field interviews were conducted with practicing systems engineers; their findings present information concerning important systems engineering issues. The overall needs conclusions are presented in the final section.

# 2. System Engineering Needs

This section presents synopses of several references that investigated the problems with mission-critical systems development and maintenance. While the references have an obvious software flavor, this is because of the increasing software content of mission-critical systems. Upon detailed review of these references, we found that many, if not most, of the critical software problems were actually *systems problems*. Indeed, the Commander of the Air Force Systems Command has stated that software is his primary problem in the systems development. [AFS89]

## 2.1. Needs Survey

### 2.1.1. 1975 NRL Navy Software Development Problems Report

"The MUDD Report: A Case Study of Navy Software Development Practices" [WEI75] is a result of a year-long investigation into Navy software problems. The report is based on interviews with more than 30 people associated with Navy software development in more than ten Navy projects.

The report chronicles the development of a mythical software system, MUDD, and describes where and how the developers went awry. The pitfalls described typify problems which actually occurred in software development efforts. The report concludes with a list of 12 recommendations designed to help those responsible for the creation and support of software systems to avoid these pitfalls. Most of the recommendations in one way or another are concerned with interfaces: interfaces between and within systems, interfaces between people, interfaces between the Navy and its contractors, interfaces within the Navy, etc.

1. *Unify life cycle control of software.* Development responsibility for a system should not be split, and maintenance activity should not be independent of development activity.
2. *Require the participation of experienced software engineers in all system decisions.* This is especially crucial for early decisions such as the determination of system configuration, assignment of development responsibility, and choice of support software.
3. *Require the participation of system users in the development cycle from the time requirements are established until the system is delivered.* Changes which are inexpensive and easy at system design time are often extremely expensive and difficult after the software has been written.

4. *Write acceptance criteria into software development contracts.* This will help avoided unnecessary misunderstandings and delays for negotiation before a system is delivered.

5. *Develop software on a system that provides good support facilities.* If necessary, consider developing support software prior to or in conjunction with system development.

6. *Design software for maximum compatibility and reusability.* Premature design decisions should be avoided; logically related systems should have their differences isolated and easily traceable to a few design decisions.

7. *Allocate development time properly among design, coding and checkout.* Since manpower-allocation estimates are based in part on the time estimates for the different phases of development, improper estimation can be quite expensive.

8. *List, in advance of design, all areas in which requirements are likely to change.* This can be done at the time requirements are stated and will help the designer partition the software to isolate areas most likely to change.

9. *Use state-of-the-art design principles, such as information hiding.* Principles which optimize reliability, cost reduction, and maintainability should be emphasized.

10. *Critical design reviews should be active reviews and not passive tutorials.* Sufficient time must be allowed to read design documents before the review, and the documents must be readable.

11. *Do not depend on progress reports to know the state of the system.* Programmer estimates are typically biased; milestones are a more accurate indication of development progress.

12. *Require executable milestones that can be satisfactorily demonstrated.* Milestones demonstrating system capabilities that will rest on major design decisions should be written into development contracts.

Regarding recommendation #9: The temptation to optimize for efficiency should be avoided. If not, the developed systems will tend to be expensive, late, unreliable, and difficult to improve or maintain. The application of information hiding is especially critical with respect to areas where the requirements are most likely to change (e.g., interfaces with other systems over which the developers and users have no control).

Regarding recommendation #10: Alternative design decisions and the reasons for eliminating them should be discussed. In addition, no code should be written until the design is approved. This is especially crucial since the cost of a design change during coding may be 2 or 3 times the cost of the change before coding, and the multiplier becomes larger the farther the system progresses in the development cycle.

### 2.1.2. 1979 GAO Report on Problems with Software Contracting

"Contracting For Computer Software Development—Serious Problems Require Management Attention To Avoid Wasting Additional Millions" [GAO79] is a United States General Accounting Office (GAO) report which discusses the problems that Federal agencies have encountered in contracting for computer software development and recommends means for improving such contracting.

> GAO sent questionnaires to 163 software contracting firms and 113 Federal project officers who had experience with software development contracts to identify what had caused trouble and what might be done to improve development efforts. Certain things causing problems for both contractors and agencies were common to all reviewed contracts that had trouble.
>
> GAO examined nine cases of software development in detail. Eight had problems, but their overall performance cannot be taken as representative—some came to GAO's attention because they were failures. Nevertheless, the cases illustrated many of the same causes of difficulty that GAO's questionnaire respondents had identified.
>
> Only one of the nine cases yielded software that could be used as delivered. The combined total costs and development times of the nine cases increased from estimates of $3.7 million and 10.8 years to an actual cost of $6.7 million and an actual duration of 20.5 years. [GAO79]

The GAO found that "agency staff connected with software development contracts typically have little guidance, either from central agencies or from their own agency headquarters." In addition, several common causes of failure were distilled from the questionnaires and case studies:

- Agencies overestimate the stage of system de,elopment they have reached before they contract.
- Failure to specify what constitutes satisfactory performance.
- Agencies do not manage software development contracts during execution.
- Agencies do not adequately inspect and test software.
- Agencies fail to establish a single focal point for the contractor.

The GAO recommended that the National Bureau of Standards and the General Services Administration issue specific guidelines to assist Federal agencies in recognizing and dealing with the unique problems of contracting for software development. The GAO also recommended that Federal agencies which contract extensively for software development train project managers in appropriate software, contracting, and management skills.

### 2.1.3. 1984 CODSIA Report on DoD Management of MCCR

"DoD Management of Mission-Critical Computer Resources" [COD84] is the result of a task to study the issues surrounding a proposed Instruction Set Architecture (ISA) standardization policy for embedded computers. The task was performed by the Council of Defense and Space Industry Associations (CODSIA) Task Group 13–82 for the Under Secretary of Defense, Research and Engineering. After a preliminary evaluation, the CODSIA Task Group felt that its scope was too limited, and that a broader investigation of DoD management of mission critical computer resources (MCCR) in the 1980s was more appropriate.

The Task Group membership reflected all the major segments of the industry, the goal being to develop an industry consensus. Thus, it is believed that the report provides a balanced industry position on DoD's management of MCCR.

> In its efforts to come to grips with the complex concerns and conflicts surrounding DoD computer resource (CR) management policy, the CODSIA Task Group evolved 12 major issues which, it was generally agreed, address most of the concerns expressed by all the interested parties. Overall, it was agreed that six of the issues address problems faced by the Military Services as they apply computer resources to meet operational objectives—high development cost and risk, high operational/logistics cost, low operational availability, poor battle survivability, and so forth. Solutions to these problems, at least in the near term, all seem to require standardization in some form. This thrust toward standardization leads to the remaining six issues, which address the problems standardization can bring to the vital need to maintain technological leadership in mission critical systems, and to the requirement by law that DoD maintain competition in its procurement practices. [COD84]

The 12 issues identified by the Task Group are enumerated below. In some cases, the issues have been synthesized with the group's associated recommendations and/or agreed upon causes. A summary of the Task Group's central or recurring recommendations are provided following the list.

1. Expensive and lengthy system development and evolution is the result of fluctuations in funding, changes in mission and threat, and unplanned/unforeseen (i.e., non-adaptable) advances in technology.
2. The high cost and risk for non-transportable software is due to the uncontrolled proliferation of computer architectures, run-time environments, and programming languages.

3. Operational availability and survivability are critical mission requirements which need to be addressed by higher level standards, commercial involvement, and a new approach to CR management.
4. The high cost and difficult logistics support for mission critical systems may benefit from commercial production and logistics support capabilities.
5. Solutions to the mobilization of the U.S. commercial CR industry may lie in the commonality between military and commercial products of the future, as well as higher level standards and consideration of technology insertion requirements.
6. Clearly stated service-wide and long-term operational objectives and priorities are required to meet the wide range of needed mission critical systems.
7. The DoD must take steps to encourage private (vs. government-funded) investment, innovation and development of MCCR technology.
8. DoD management of the procurement of ISAs and computer products needs to become more flexible in order to resolve conflicts of interest outside the Department.
9. Technology insertion, made difficult by rapid product obsolescence, complex real-time applications, etc., requires a strong and central oversight and integration of DoD initiatives such as STARS, VHSIC, and Ada, and higher level, vendor and technology-independent standards which are consistent, unified, and have built-in flexibility and adaptability so that applications can be adjusted to continual technology change.
10. DoD (vs. voluntary) funding and support of standards processes are required to insure that the resultant standards are responsive to the DoD's needs.
11. The DoD should continue to review and update its CR acquisition practice in order to preserve competition and innovation in the defense industry.
12. The Office of Secretary of Defense (OSD) should continue to review and update CR policy concerning the management of computer resources and initiatives, keeping in step with continual technology advances.

The Task Group's recommendations define two major thrusts: new and improved CR acquisition and management standards and policies which reflect a long-term and broad-based perspective, and increased emphasis on the use of the commercial industry and the application of commercial technology and products towards MCCR needs.

### 2.1.4. 1984 IDA Report on DoD Related Software Technology Requirements, Practices, and Prospects

"DoD Related Software Technology Requirements, Practices, and Prospects for the Future" [RED84] is an Institute for Defense Analyses (IDA) report on the current DoD practices and approaches to software development, and future prospects for DoD related software technology. The report contains an analysis of eight systems; six major defense systems, one from NASA and one commercial. Of the eight systems examined, five of the development efforts are considered to be 'successful.' No major differences were discovered between the DoD developments and the NASA or commercial developments.

The report includes th‿ following characterization of DoD mission-critical systems and the current state-of-the-practice:

- large-scale, real-time, and fail-safe operation
- long life with continual change
- development by large team and maintenance by a different organization
- co-existence with older systems and interfacing with unique hardware

> The current state of practice is experiencing problems in meeting current requirements. Programs have difficulty defining requirements and requirements are constantly changing as software becomes increasingly responsible for implementing new functions. Other problems are in budgeting, staffing, scheduling, and with product quality. [RED84]

In fact, the report states that "[t]he single overwhelming commonality that existed among the systems investigated was the requirement to accommodate change." Several additional common software development problems were identified:

- Few systematic techniques and little automated support existed for the requirements definition process. Even the tracing of requirements into the design, code, and test is predominantly conducted manually.
- Estimates of software cost, staffing requirements, and schedules are frequently inaccurate. The size of the development effort is usually underestimated, and the productivity of personnel overestimated.
- The level of discipline and formality with which software technology is applied on development efforts is lacking. There is a deficiency in automation as well as in identifiable and formal methods.

- Software is often not treated as a manageable entity. "This may stem from a lack of understanding on the part of management and a general feeling of discomfort when dealing with software[!]"
- There is a lack of generally accepted quantitative measures for software development and engineering.

### 2.1.5. 1984 TRW Productivity Study

"A Software Development Environment for Improving Productivity" [BOE84] is a paper which describes the beginning stages of a decade-long and continuing TRW corporate initiative aimed at improving the productivity of their software development efforts.

In 1980 TRW conducted an extensive software productivity study of corporate objectives, requirements, and alternatives. The study included an internal assessment of their software development practices, as well as an external assessment of several industrial and academic organizations with experience or active R&D programs on software support environments. More specifically, the study included an analysis of the requirements for a company-oriented software support environment, an evaluation of the current and future supporting technologies, and an economic analysis for justification of corporate investment of software productivity aids.

The primary conclusion from the software productivity requirements analysis was that significant productivity gains are possible, but they require a long, sustained effort of integrated program initiatives in several areas. High payoff productivity increases of 39 to over 100 percent improvements were reported in two surveys and measured on a medium sized (roughly 17,000 DSI) software development effort. Several additional, complementary conclusions were also made:

- *Immediate access to a good set of integrated software tools has the highest payoff.* Other productivity aids include personal terminals, private offices, on-line documents, and electronic mail.
- *Office automation and project support capabilities are required for all project personnel.* These are among the most often used tools, crossing project assignments.
- *There is high payoff in placing all software development artifacts on-line and providing tools to support easy access to them.* This is costly, but well worth the expenditure.

- *User interface standards are essential for preserving the conceptual integrity of an evolving support system.* In addition, utilities which embed these standards are an excellent means of implementation.
- *User acceptance of novel development environments is a gradual process that requires careful nurturing.* Strong user involvement, training, and documentation are highly recommended.
- *Local area networks strongly support distributed work environments.* A LAN coupled with electronic mail allows a physically scattered group to work effectively as a team.
- *Private offices improve productivity.* Each technical staff member should be provided a private office, complete with a networked terminal or workstation.

The principle author of this paper, Barry Boehm, authored another paper a few years later [BOE88] concerning the spiral model of software development and enhancement. Reportedly, TRW continues to develop, refine, and apply the spiral model to the research and development of an integrated software development environment as recommended above. The paper discussing the spiral model is also included (below) as part of the systems engineering needs survey.

#### 2.1.6. 1987 Paper on the Essence and Accidents of Software Engineering

"No Silver Bullet: Essence and Accidents of Software Engineering" [BRO87] is a much-referenced paper by Fred Brooks on the *essential* (i.e., inherent) and *accidental* (i.e., nonessential) difficulties of software engineering. Brooks' position is that there is no *silver bullet*, i.e., "something to make software costs drop as rapidly as computer hardware costs do."

There are four inherent difficulties of software development: complexity, conformity, changeability, and invisibility. "Software entities are perhaps more complex for their size than any other human construct..." In addition, software is typically made to conform to other system elements, and these changes are forced upon software on a continuous basis. Finally, software is not only invisible; it cannot be visualized "The reality of software is not inherently embedded in space. Hence, it has no ready geometric representation in the way that land has maps, silicon chips have diagrams, [or] computers have connectivity schematics."

The three most significant software technology breakthroughs of the past are: high-level languages, time-sharing, and unified programming environments.

These breakthroughs have each solved accidental difficulties, that is, not essential difficulties. More recently, there have been several technical developments which are most often proposed as potential silver bullets: Ada and other high-level language advances, object-oriented programming, artificial intelligence and expert systems, automatic programming, graphical programming, program verification, environments and tools, and workstations. Most of these developments have limited potential, and are solutions to the accidental difficulties of software development. The exception is expert systems. Expert systems may provide the experience and accumulated wisdom of the best software engineers to less-expert engineers—"no small contribution."

As for the future, four new potential silver bullets seem very promising: buy versus build, requirements refinement and rapid prototyping, incremental development, and great designers. Unlike the potential silver bullets discussed above, these solutions attack the essence of the software problem, i.e., the formulation of complex conceptual structures. "The most radical possible solution for constructing software is not to construct it at all." Buying applicable off-the-shelf software packages should become increasingly easier as more and more vendors offer more and better software products.

In addition, the development of approaches and tools for rapid prototyping is a most promising solution to the unavoidable iterative specification of requirements:

> The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.
>
> Therefore, the most important function that the software builder performs for the client is the iterative extraction and refinement of the product requirements. For the truth is, the client does not know what he wants. The client usually does not know what questions must be answered, and he has almost never thought of the problem in the detail necessary for specification. ... So in planning any software design activity, it is necessary to allow for an extensive iteration between the client and the designer as part of the system definition. [BRO87]

As for software construction itself, incremental development (e.g., the spiral model) is advocated, as opposed to the *writing* or *building* of software in its entirety. At a very early stage, the system should be made to run, and at every stage in the process, there should be a working system. In this fashion, the software system can be systematically *grown*.

Finally, software organizations should commit themselves to acquiring and growing top-notch software personnel. "The differences between the great [software designers] and the average approach an order of magnitude." Thus, "great designers are as important to [a software organization's] success as great managers are, and...they [should be] similarly nurtured and rewarded."

## 2.1.7. 1988 IDA Report on Concurrent Engineering

"The Role of Concurrent Engineering in Weapons System Acquisition" [WIN88] is the result of an IDA investigation into the claims—made by several U.S. companies—of improved product quality at lower costs and shortened product development times through the use of *concurrent engineering*. The report is an assessment of these claims. In carrying out their task, the IDA study team reviewed published papers, visited and held technical discussions with fourteen major U.S. corporations, and conducted workshops.

The basic principle of concurrent engineering is the *simultaneous design* of a *product* and its associated down stream *processes*:

> Concurrent engineering is a systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule, and user requirements. [WIN88]

Concurrent engineering is realized through the modification of management, engineering, and business processes to include a more systematic method of concurrently designing both the product and the downstream processes for producing and supporting it. The primary motivation behind concurrent engineering is the desire—at the corporate level—to *improve quality*. Companies are beginning to regard quality as the driver for achieving lower costs and shorter schedules. [This is evident in one major DoD contractor's battle cry: "Quality first, cost and schedule will follow."]

The IDA study team identified three complementary classes of activities among the concurrent engineering initiatives:

- **engineering-process initiatives** — management actions to improve the organization and the procedures used to develop a product
- **computer-based support initiatives** — the improvement of computer-based tools, database systems, and special purpose computer systems in order to better support product design, production planning, and production

- **formal methods initiatives** — an improved utilization of statistical, experimental, and quality engineering techniques—including automated methods—for managing complex system trade-offs and for finding optimum design and production process parameters

A common engineering-process initiative is the formation of multi-disciplinary design teams (i.e., marketing, production, engineering, manufacturing, support, purchasing, and other specialist), and their early and continued involvement in the design process, identifying potential problems and finding timely solutions. Other engineering-process initiatives include an emphasis on customer needs and quality improvement, and on promoting employee involvement in continued process improvements.

There are two common computer-based support initiatives or goals: 1) a shared information base which would be used as the common source for most activities, and 2) an integrated environment in which computer-based tools and software of varied sources could be efficiently used in cohesion. Organizations are striving for a computer-based support environment which promotes communication and the unification of product life cycle activities across disciplines.

### 2.1.8. 1988 TRW Paper on the Spiral Model

"A Spiral Model of Software Development and Enhancement" [BOE88] is a paper by Barry Boehm, TRW Defense Systems Group, which advocates a new software process model. "The spiral model of the software process has been evolving for several years, based on experience with various refinements of the waterfall model as applied to large government software projects." The spiral model is one result of a growing concern that traditional software process models were discouraging more effective approaches to software development (e.g., prototyping and software reuse):

> A primary source of difficulty with the waterfall model has been its emphasis on fully elaborated documents as completion criteria for early requirements and design phases. For some classes of software, such as compilers or secure operating systems, this is the most effective way to proceed. However, it does not work well for many classes of software, particularly interactive end-user applications. Document-driven standards have pushed many projects to write elaborate specifications of poorly understood user interfaces and decision-support functions, followed by the design and development of large quantities of unusable code. [BOE88]

Basically, the spiral model consists of a sequence of steps which define a cycle, phase, or "round" (i.e., circuit of the spiral):

1. **Analysis** — A preparatory analysis of objectives, alternatives, and constraints which are used to drive the current round.
2. **Risk Management** — The evaluation of alternatives, and the identification and resolution of risks. This may include prototyping, simulations, models, benchmarks or other forms of risk resolution.
3. **Development** — The development of the current round's product(s).
4. **Verification** — The verification and/or validation of the current round's product(s).
5. **Planning** — The planning of the next round, including the approach, budget, scope, schedule, and resources.
6. **Review and Commitment** — The formal review of all the current round's activities and products, and of the plan for the next round. Upon successful review and agreement, a commitment is made to perform the next round.

"The spiral model reflects the underlying concept that each cycle, or round, involves a progression that addresses the same sequence of steps, for each portion of the product and for each of its levels of elaboration, from an overall concept of operation document down to the coding of each individual program." Nominally, each iteration, or round, delves deeper into the project development (e.g., from concept of operation to requirements to design to detailed design and implementation). However, the model is extremely flexible. For example, it allows "go-backs," i.e., a redirection of development effort to perform some rework, perhaps from a previous round. Furthermore, the risk-driven analysis of alternatives may at any time redirect the development effort in terms of rework, step sub-setting, or specifying/adjusting design-to-cost levels-of-effort.

"The major distinguishing feature of the spiral model is that it creates a *risk-driven* approach to the software process rather than a primarily *document-driven* or *code-driven* process." Previous software development projects have suffered from a lack of emphasis on risks and risk management, and also from pursuing stages in the wrong order. By performing great elaboration of detail for the well-understood, low-risk elements, and little elaboration of the poorly understood, high-risk elements, projects often head for disaster while it appears they are making progress.

As a result, projects have often "come to grief" through the development of hard-to-change code, before addressing long-range architectural and usage considerations, and other critical issues. This is called "information sclerosis"—the solidification of inappropriately developed software which becomes increasingly hard to modify. For example, information sclerosis may be caused by the use of *temporary* work-arounds which ultimately become the basis for subsequent decisions and further development. Another common cause for information sclerosis is the over-emphasis of performance optimization.

One of the main advantages of the spiral model is that it focuses on eliminating errors and unattractive alternatives early. To achieve maximum benefit, the followiیg supplementary actions should be taken when applying the spiral model:

- software quality objectives should be specified in the analysis step, promoting quality software development
- the major sources of change should be included in the objectives (the objective being the ability to accommodate such changes), promoting information hiding and reducing the chances of information sclerosis
- steps involving the identification and evaluation of alternatives should include the early focusing of attention on options involving the reuse of existing software
- prototypes are valuable and practical risk-reduction tools, and should be utilized whenever appropriate, including the later rounds of the project development effort

Ironically, one of the disadvantages of the spiral model is its flexibility. In terms of contracting and acquisition, it presents challenges for accountability and control (e.g., specifying deliverables). The spiral model also needs further refinement and elaboration. However, some of its advantages can be easily adapted to other process models:

> Efforts to apply and refine the spiral model have focused on creating a discipline of...risk management, including techniques for risk identification, risk analysis, risk prioritization, risk-management planning, and risk-element tracking. ...
>
> Even if an organization is not ready to adopt the entire spiral approach, one characteristic technique that can easily be adapted to any life-cycle model provides many of the benefits of the spiral approach. ...[T]he Risk Management Plan...basically ensures that each project makes an early identification of its top risk items, develops a strategy for resolving the risk items, identifies and sets down an agenda to resolve new risk items as they surface, and highlights progress versus plans in monthly reviews. [BOE88]

DoD–STD–2167A and AFR 800–14 both require that developers produce and use risk management plans. This may provide a foundation for tailoring spiral model concepts into the more established software acquisition and development procedures.

## 2.1.9. 1989 AF Report on Adapting Software Development Polices to Modern Technology

"Adapting Software Development Policies to Modern Technologies" [AFS89] is an Air Force report by the Committee on Adapting Software Development Policies to Modern Technology (chaired by Walter Beam) of the Air Force Studies Board (AFSB). The committee was tasked to investigate software problems, and their solutions. In addition, the committee was to investigate why other recent studies on the subject have not resolved the problems with software development. Of special interest were "newer methods of software development now being introduced for large, high-technology systems," and the possible inadequacies of "conventional software acquisition techniques and policies."

The general conclusion on why software problems within the Air Force still exist is that even though some (mostly organizational) steps have been taken, the demands being placed on software are increasing faster than efforts to address the problems. The report contains a significant amount of analysis of AF-specific considerations (e.g., organization, policies, regulations, directives). However, the report identifies three general strategies which are potential solutions to the Air Force's software problems:

- risk reduction
- improved acquisition and development processes
- strengthening personnel resources

**Risk Reduction**: Efforts should be made to identify and control risks as early as possible in systems and software development. For example, policies should mandate the use of risk management plans.

Also, unprecedented systems—"systems for which there has been no precedent in the form of similar systems or systems performing the same functions, or for which the design teams lack full or applicable system design experience"—should receive special attention with regards to risk reduction. At the acquisition level, for example, unprecedented systems should require a successful demonstration/validation phase prior to entering a full-scale development phase.

In addition, process models other than the waterfall model should be considered as a means of managing the risk inherent in unprecedented systems:

> The waterfall model is satisfactory *only* for precedented systems, i.e., those for which substantial implementation experience exists. In other cases, it is highly risky and does not emphasize safeguards in the definition of

requirements, evolution of system design, and development or maintenance of effective design teams. *In short, alternative models are needed for the development of unprecedented systems.* [AFS89]

Other risk reduction related strategies to eliminating software problems include an enhanced product quality emphasis, and a technology advancement awareness. In the past, product quality was merely addressed by testing activities. Quality assurance, imposed from the start and throughout software development efforts, is needed to reduce the risk of failure. Software metrics and quality evaluation research and development should provide continued risk reduction capabilities. Numerous other software-related technologies are rapidly advancing and should be funded and/or monitored for potential impact and insertion into the Air Force arena (e.g., higher level languages, executable specifications, software reuse).

**Improved Acquisition and Development Processes**: There is a need for innovative acquisition *tailoring* and alternative development processes such as incremental development and prototyping. For example, "user involvement should be tailored for each program, varying from cases requiring very limited involvement to ones in which a user will assume the lead role." From user involvement to development process models, it is acknowledged that no single acquisition strategy can possibly serve all situations.

Software engineering environments (SEEs) are strongly advocated as a means to improve the software development process. In addition to the more common software development tools, integrated SEEs should provide capabilities for design exploration (design analysis tools, reusable software, simulators, VHLL support, executable specification, quality assessment), management support, and communications (especially for organizationally or geographically distributed design teams).

Other recommendations include using the designated maintenance agent for IV&V during software development, and the avoidance of fixed price contracting in acquisition of unprecedented systems.

**Strengthening Personnel Resources**: The skill, experience, and communication ability of management and technical personnel are critical to successful software development. A common problem with systems development is the dichotomy between systems and software engineers. Systems engineers (and managers) need a better understanding of software, and software engineers need a broader understanding of systems.

It is recommended that the Air Force take several actions to strengthen their personnel resources, including: training and educational programs, the formation of senior-level engineering advisory teams, and changes in policy which would motivate personnel (e.g., advanced pay scales and promotion schedules).

### 2.1.10. 1989 DARPA Report on their Initiative in Concurrent Engineering

"DARPA Initiative in Concurrent Engineering (DICE) - Phase 1" [SIN90] documents the first phase of a (roughly) five-year, 100-million-dollar DARPA initiative in concurrent engineering that was begun in 1988. The initiative is a very broad yet focused research, development, and demonstration effort in the area of mechanical and electrical design and manufacturing.

The program is structured around the concurrent engineering of a jet aircraft engine. The General Electric Aircraft Engines (GEAE) CF6-80C2 engine was chosen by GEAE designers as the demonstration problem. The design and manufacture of aircraft engines involves both aerodynamic (engine flowpath) and mechanical (engine blading) engineering considerations. Previous aircraft design processes would typically approach the problem sequentially, addressing aerodynamic and then mechanical/structural issues, and iterating over the process steps during periods of analysis and refinement.

> By designing from a system or global viewpoint, engineering productivity improves due to integrated/automated analysis procedures, and a multi-disciplinary approach increases knowledge transfusion among the designers. A concurrent design of flowpath and blading with simultaneous consideration of both aerodynamic and mechanical design criteria under an integrated design environment demonstrates the methodology.
>
> The goals of this project is therefore to develop the methodology and tools for concurrent design of engine flowpath and blading, leading to significant increase in engineer productivity and superior product quality. [SIN90]

As part of the effort, a consortium of university/industry has been formed to research concurrent engineering issues. The consortium—under the program management of GEAE—includes GEAE, West Virginia University (WVU), Cimflex Teknowledge Corp., General Electric Corporate Research and Development (GECRD), Carnegie Mellon University, Martin Marietta Laboratories, Howmet Corporation, Rensselaer Polytechnic Institute (RPI), University of California at Santa Barbara, and North Carolina State University. In addition, a Concurrent Engineering Research Center (CERC) has been established at West Virginia University to showcase the DICE research and to transfer the technology to industry.

The major tasks of the DICE initiative—besides the establishment of the CERC—are to develop a generic computer-based concurrent engineering architecture for the design and manufacture of structural components and electronic assemblies; to develop new, domain-specific, concurrent engineering methods and tools (and integrate them with those that already exist) for the development of structural components and electronic assemblies; and, to demonstrate the concurrent engineering technology by manufacturing (increasingly complex) structural components from composite materials and electronic assemblies utilizing the DICE architecture, methods, and tools.

The DICE definition of concurrent engineering is seemingly overambitious. Not only does it account for the constraints and requirements of downstream disciplines, it is said to be "a revolutionary approach to simultaneously conduct research and development, design, and manufacturing of [structural] components and [electronic] assemblies..." Others have warned of the inherent problems with simultaneous *design* and *manufacture*. [WIN88] It is possible that the scope of the DICE initiative allows such an approach, especially considering the wealth of information available from previous aircraft engine systems. However, there are other aspects of their concurrent engineering research which are more conventional. For example, the DICE effort exhibits an emphasis on integrated environments, information sharing, life cycle quality assurance, life cycle cost analysis, and the design for manufacture, assembly, and testability. In addition, the DICE project has given special consideration to multiple engineering disciplines, and multiple engineers, working simultaneously on the same design (conceptually and physically).

Of the work carried out so far, there are four areas which are of particular interest at this point: information content and flow, information management data base, architecture definition, and the workstation node prototype.

**Information Content and Flow:** After conducting meetings with other DICE team members and practicing engineers, and surveying technical literature concerned with concurrent engineering, the GEAE task group on information content and flow have made the following four recommendations:

- Involve engineering users in the definition of engineering processes. This allows users to develop a greater appreciation for the discipline required to manage engineering data, and its payoff.
- Engineering data management systems must be flexible to accommodate the ever changing nature of the design process.
- Data definitions of shared data must be provided to the users of the data, as well as the ability to quickly reference desired data.
- Random access of engineering data maximizes the value of the data to the engineer.

**Information Management Data Base:** The RPI task group on the information management database is developing an object-oriented database system to support concurrent engineering. The system is based upon a previous object-oriented engineering database system called ROSE. However, the new system is programming language independent, whereas ROSE was tied to a ROSE-specific prototyping language. The new system, called ROSE-IC, is based upon Objective-C, and has a powerful Objective-C programming environment. Other versions may soon be implemented as well, including C++ and CLOS (Common Lisp Object System).

ROSE-IC has also been injected with concurrent engineering concepts. For example, the new version of ROSE represents designs as sets of objects stored in files. These files can be transferred between a file server and a workstation before and after an engineering design session. This allows the database system to achieve a high level of performance, but with the penalty of allowing conflicts. A scripting utility supports the concurrent editing of objects, where scripts are used to capture design changes made by individual users. Additional research is being applied to various protocols for controlling the scope of potential conflicts which may result. The system also supports a graphical user interface, an SQL interface to a relational database, and tools to edit and manipulate its databases. A meta-tool which generates design data conversion tools is also under development.

**Architecture Definition:** The WVU task group on architecture definition is working on a three Cs model of a concurrent engineering architecture; a collection of connected workstations linked in a network with computer assistance for Communication, Cooperation, and Coordination. On every networked computer there is a layer of software which enables the engineers "to take advantage of cooperating experts and their tools on the network."

Confusingly, **Communication** is supported by a software system called *Cooperate,* which "connects a group of engineers and organizes a virtual 'meeting' on the network." Another utility called *View* enables the communication of graphics during the meeting. Work is also being performed on object-oriented communication, where objects in different computers may invoke each other's behavior.

**Cooperation** is supported by a base layer of systems software called the *Concurrency Manager* (CM), which resides in each workstation. The CM takes advantage of the assumed permanent inter-process communications between the workstations, thus allowing any designer to invoke the CM in their workstation to communicate with any other application or designer in the network.

**Coordination** is aimed at making the activities of the designers and the project leader visible to each other. This is accomplished through the use of a globally visible blackboard workspace. Through the blackboard, the project leader is able to assign tasks, place portions of the design (from the engineering database) on the blackboard, and coordinate any proposed changes to the design.

At the time of writing (of the DARPA report), these systems were not yet fully developed, and were scheduled for completion in mid-1990.

**Workstation Node Prototype**: The GECRD task group on the workstation node prototype is investigating the integration of the DICE information architecture framework. This task group has made two major conclusions:

- the spreadsheet paradigm of user interface is appropriate and comfortable for design engineers, and is better accepted than the Macintosh-style of menu/icon interfaces
- the wrapper concept is relatively simple to implement, holds good potential for the automation of wrapper generators, and may be a cost-effective alternative for the application classes in question

Wrappers enable non-DICE applications to access data and data services within the DICE environment in an integrated manner. A wrapper must provide a standardized interface for an application's integration; it must also surround an application such that the interface to the application itself remains unchanged. More specifically, wrappers must: facilitate bi-directional transfer of data and directives/status; present data to the user in a form which is easy to comprehend; hide the internal workings of the application from the outside world; provide seamless access to DICE kernel services; and provide a mapping between the application data and the DICE object space.

The integration environment for the workstation node prototype included a spreadsheet with access to engineering data, an optimization package, a namelist oriented method and DICE kernel services. The spreadsheet interface was used to reference the engineering data, experiment with the data, and optionally publish results. The system contained a suite of wrappers, including a host wrapper, a user-interface wrapper, and separate tool wrappers.

Under this prototype integration environment, the task group's recommendations seem reasonable. However, under the much broader scope of systems engineering, it is not likely that such an approach would be feasible.

## 2.1.11. 1989 Requirements Engineering and Rapid Prototyping Workshop Proceedings

"TTCP Requirements Engineering and Rapid Prototyping Workshop Proceedings" [BLA90] documents the findings of a two-day workshop hosted by CECOM's Center for Software Engineering in November of 1989. The workshop was sponsored by The Technical Cooperation Program's (TTCP) Panel on Software Engineering.

The workshop had 49 participants (from the U.S. and abroad), most being leading requirements engineering experts. They met to share information, to identify and clarify the most pressing issues, and to provide recommendations to DoD for management, development, and research relating to requirements engineering. The workshop was organized into three independent working groups by the following topics: the requirements engineering process; requirements engineering methodologies, tools, and languages; and knowledge-based techniques and rapid prototyping.

The DoD's software production costs have been growing exponentially. This is due partly to the large number of errors introduced during requirements definition, in conjunction with the high cost of software error correction in the later stages of software development. It is believed that proper (i.e., systematic, disciplined) requirements engineering would alleviate this situation:

> As much as fifty-five (55) percent of system errors are introduced during the requirements definition phase. ...
>
> For commercial and military computer-based systems alike, experience has shown that, especially for large and complex system developments, it is rare that the true needs of all stakeholders are fully stated and understood from the outset. Furthermore, even the requirements that are understood are not always agreed upon by all parties. To complicate matters more, requirements that have been documented are sometimes subject to interpretation by both users and developers. In addition to these problems, once requirements have been baselined, there are difficulties associated with anticipating, controlling, and managing changes to the baseline. [BLA90]

The working groups each identified several major issues concerning requirements engineering within the DoD. The following list is a highlight of these issues:

- uncertainty and change are difficult to cope with
- validation of requirements is critical to project success
- multiple stakeholders make it difficult to reach closure

- a method to track progress in requirements development
- a lack of widespread awareness of the importance of requirements engineering, especially in management and acquisition offices
- a lack of emphasis for the requirements engineering process throughout the life cycle, and for its related policy and funding support
- an unawareness that requirements engineering is vital to system success, and hence to national security and economic vitality
- languages and methods fail to capture requirements information
- a lack of understanding of non-functional requirements
- requirements engineering tools are not integrated
- there is a lack of understanding of what to measure and how to measure key requirements process parameters

The working groups each made several recommendations. Of particular importance are those recommendations which were common among the working groups, which operated independently. The recommendations mostly call for additional research and development:

- DoD policy changes (e.g., supporting incremental or evolutionary acquisition, and a requirements-centered development life cycle model) with corresponding government acquisition personnel training
- additional emphasis and exploration of requirements validation (e.g., mandating a requirements validation plan for every project, executable specifications, validation of non-functional requirements)
- methods and techniques for measuring requirements related attributes and progress in the requirements engineering process
- the specification of non-functional requirements (e.g., conflict resolution, quantification, inter-relations, impact analysis)
- requirements trade-off analysis (e.g., capture, organization, and evaluation)
- requirements traceability (e.g., a life cycle requirements database to capture and manage attributes of individual requirements and to provide traceability between prototypes and systems under development)
- the communication between multiple stakeholders, and the difficulties in reaching closure

## 2.1.12. 1989 House of Representatives Study on Problems in Government Software Development and Regulation

"Bugs in the Program: Problems in Federal Government Computer Software Development and Regulation" [REP89] is an inter-committee Congressional report which "is intended to alert [Congress] to the growing problems the Government faces as its dependence on computers increases." The report identifies three problems in the area of computer software:

- software procurement
- software assurance and regulation
- maintaining trained, professional personnel

**Software Procurement**: The current policy needs to be reformed with an emphasis on systems engineering principles:

> ...[I]t is a theme of this report that software cannot be properly developed using the welter of regulations presently in force. While software now drives system requirements, the procurement system still focuses attention on hardware. ...[N]ew software-oriented procurement regulations reflecting today's reality in buying systems [are needed]....
>
> Critical to the new procurement statute must be a 'system-first' perspective. It is important that consideration for system requirements drive managers. Allowing hardware and software development to proceed separately or in isolation is a formula for problems. Giving software equal status in planning for procurement will certainly change Government procurement. Good systems engineering, where the program manager factors in user needs, safety and security at the outset of design and seeks trade-offs to match his available resources, may leave nothing tangible to show when the time comes for the next budget cycle. No program manager relishes the thought of defending a request for funds when the major activity seems to be endless arguments over abstruse technical points by large numbers of well-paid engineers. Yet experience shows that this is precisely the course to follow because it answers most, if not all, the questions that are expensive to fix on a production line. ...
>
> What may finally force a re-definition of this procurement system is continued failure to balance the budget. The strange policy where the Government pays twice for a system—once to buy it and again to make it work the way it was expected—cannot be sustained in an era of multi-billion dollar shortfalls in the Treasury. [REP89]

**Software Assurance and Regulation**: With the growing application of computers, industries such as medical treatment and air transportation are becoming potentially hazardous to the public's health and safety. "Agencies charged with regulating these industries have no accepted method for assuring that computer software will operate safely in these applications."

First, the ability to measure software is needed. Ultimately, regulatory agencies need automated tools which are capable of providing quality evaluation of computer software. In addition, safety and security issues require early identification, systems-level attention, and collaborative user-involvement throughout system development.

**Maintaining Trained, Professional Personnel:** Any successful project is the result of trained, professional personnel.

The Government is in need of qualified personnel, and must adjust its recruitment and retention policies in order to compete with industry. This will be even more critical in the future.

In general, academia must resolve the issues of computer science vs. software engineering curriculum. Software development of large and complex systems requires *team* efforts, which are not yet being widely taught.

Other personnel issues which need further attention are professional ethics and, possibly, certification.

The subcommittee concluded with five specific recommendations:

1) the establishment of a Working Group on Software Development Improvement focused on solving software problems within the Government,
2) a reformed procurement policy,
3) continued and focused software assurance R&D,
4) continued basic research in computer science and software engineering, and
5) increased SEI activity/participation from Federal agencies besides the DoD (e.g., Food and Drug Administration, Federal Aviation Authority, Internal Revenue Service, Social Security Administration).

### 2.1.13. 1990 Hughes Paper Project Management and Successful Projects

"An Exploratory Analysis Relating the Software Project Management Process to Project Success" [DEU90] is an informal investigation of the management factors which are most responsible for successful software development projects. The study tests a conceptual model (of software project management) and set of hypothesis against data gathered from several software development projects. The data is analyzed and suggests that the model is feasible, providing a "partial unveiling of the 'black magic' craft practiced by skillful and experienced software project managers." Continued long-term research is planned to increase confidence in and refine the model.

> This study probes into...aspects of [project] success by characterizing the factors of adversity that may be present in the project environment and the factors of management skills that may be put forth to manage and overcome this adversity. These are then related to both project technical and cost/schedule performance factors. [DEU90]

The investigation included an examination, via survey questionnaires, of 24 projects. The projects surveyed represent 21 separate organizations, and can be characterized as either real-time embedded systems or highly user interactive information systems with stringent performance requirements. The size of the developed software systems ranged from 100,000 to 2-million LOC, with a median of 400,000.

The basic hypothesis of the study is that the degree of *management* exercised over six specific project factors has a significant positive effect on technical and business project performance. These six factors "can be viewed as the key objectives of software project management. Associated with each objective is a specific potentially corrupting adversity that must be controlled." Table 2-1 summarizes the hypotheses and includes additional hypothesized management actions that would neutralize the adversities.

Table 2-1. Success-oriented Software Project Management

| Objective | Potential Adversity | Controlling Actions |
|---|---|---|
| business risk management | business constraints (cost and schedule) | technical scope definition (i.e., clarity, scope, and stability of technical requirements; requirements prioritization; user/customer/contractor dialogue and collaboration |
| technical risk management | technology development | risk mitigation measures and risk monitoring |
| external interfaces management | external interface adversity (i.e., complexity of interactions with the surrounding environment) | interface management (i.e., provision of appropriate activities and process steps for interactions with elements external to the system) |
| multiple user need management | number of user agencies | multiple user reconciliation |
| problem scope management | project size and character (i.e., magnitude, difficulty and complexity) | quality and retention of personnel resources; quality and assignment of technical/physical resources |
| project planning, feedback, and control | uncontrolled change | strategic planning followed by tactical planning and control (i.e., monitoring, feedback, and risk control) |

It is further hypothesized that "the management process and its factors will be more significantly correlated to project performance for higher adversity projects. This reflects the need for a more complete and sophisticated management mechanism on difficult, complex systems."

Several specific management actions are given special emphasis as project success factors: quality personnel and their retention throughout the project life cycle, a central project function for technical definition and control (e.g., a systems engineering organization), technical scope definition, user/customer/contractor dialogue, external interface management, and risk management.

As far as factors which tend to negatively influence project success, two major risk parameters are highlighted (along with their corresponding controlling actions): unrealistically optimistic cost and schedule allocations, and the degree of technology development required for a project. Even with severe cost and schedule constraints, projects with clear and precise technical scope definition and prioritized requirements are able to effectively apply implement-to-schedule strategies, and achieve project success. For projects with high technology demands, better performance has been correlated with an emphasis on risk monitoring and management.

It is noted that numerous other correlations and causal relationships can be drawn from the exploratory analysis, providing many opportunities for continued research.

## 2.2 Conclusions from Needs Survey

From the needs survey, it is obvious that there are a host of problem areas in the systems engineering state-of-the-practice. As suggested and recommended throughout the references, we believe that the following systems engineering problem areas provide significant opportunities for cost-effective automation (see list below).

The last two problem areas are somewhat different from the rest. While they are very strongly emphasized throughout the references, they provide only indirect, yet still significant, automation potential. The remainder of this section will elaborate on all these problem areas and their potential automation.

- requirements
- collaboration
- interfaces
- risks
- change and complexity management
- quality engineering
- assurance
- integrated support environments and shared information
- personnel
- acquisition policies and practices

**Requirements**. From the very onset of systems development, there is a great need for improvement. Indeed, some feel that this is where it is needed the most. The following aspects of requirements engineering all represent areas for R&D and innovation:

- conceptualization, specification, formalization
- user involvement, validation

- realization, browsing, understanding, prototyping, simulations
- change and impact analysis, evolution, negotiation, change management
- allocation and synthesis

**Collaboration**. Systems engineering involves a great deal of internal and external collaboration. For example, internal collaboration (i.e., within the development organization) involves the joint effort of systems engineers and specialty engineers. External collaboration is at the level of the three primary agents: the customer(s), the contractor(s), and the user(s). Effective collaboration may be described as the cross-fertilization of interdisciplinary and inter-agency concerns. Automation such as groupware may support the following collaboration activities:

- concurrent consideration/synthesis for multi-disciplinary/agency concerns
- propagation of prioritized system qualities
- alternatives analysis, decision aids
- cross-impact of decisions across the disciplines/agents
- allocation, dissemination

**Interfaces**. The management and support of engineering activities associated with system and subsystem interfaces is a priority concern. Probably the most critical of these is the specification of interfaces. Other interface-related activities which need support as well are listed below:

- identification, realization, and allocation
- understanding, browsing, using, complying
- baselining, negotiating, change management

**Risks**. The proper management of risks is critical to the success of systems engineering. The following risk-related activities offer very significant automation potential:

- identification, realization
- planning, abatement, avoidance
- tracking, monitoring, assessing
- alternatives analysis, action, resolution

**Change and complexity management.** In any large systems engineering effort, there will certainly be change and complexity. Automation potential exists for the following aspects of change and complexity management:

- identification, linkages, interrelationships
- information fusion, abstraction, projection
- impact analysis
- change facilitation, management
- tracking, assurance

**Quality engineering.** There is a great need for the automated support of "total system quality" in systems engineering. This includes the following:

- quality specification, prioritization
- allocation, analysis
- synthesis, impacts, trade-offs
- statistical quality control

**Assurance.** Assurance is becoming an increasingly more difficult task for large and complex systems, while at the same time becoming more essential for safety, security, and trustworthiness. The following aspects of assurance offer potential automation:

- dissemination, active review, understanding
- evaluation
- reaction, response, and change as a result of assurance activities
- acceptance, inspection, integration, test

**Integrated support environments and shared information.** This is the most obvious and most directly applicable problem area identified for systems engineering; *integrated* computer-based systems engineering support environments and *shared* information. It is also directly in line with our work.

**Personnel.** There is undoubtedly an overwhelming consensus in the literature that good quality personnel is the most essential aspect of systems engineering. Although this is not directly an automation issue, it may still provide a guiding light for other automation rationale. As an analogy of the response time vs. productivity argument for interactive computing tools, an effective systems engineering environment will act as a systems engineering catalyst in a similar manner.

The net result is to make each person involved in the systems engineering process (who may use the automated, integrated environment) more productive than they would be otherwise. As a result, non-experts could be raised to the level of experts, and experts may be made even more productive.

**Acquisition policies and practices**. Another overwhelming consensus that is not directly related to automation are the problems with acquisition policies and practices. Systems engineers (and other personnel involved in systems engineering) may benefit greatly from the automated support of acquisition/contractual information, regulations, DIDs, and policies. In practice, many contractual guidelines and/or details are not carried out, or adhered to. These guidelines are generally specified in great detail for good reasons, yet very often are not implemented (i.e., lost in the shuffle), to the detriment of the project.

# 3. Field Interviews

To ascertain system engineering needs, field interviews were conducted with practicing systems engineers. This section presents information concerning the interview methodology and the field interview results. Appendices A - E of this document contain the actual findings from interviews conducted at Rome Laboratory, NADC, and IBM.

## 3.1. Field Interview Methodology

The following sections describe the objectives and methods applied in the field interviewing process.

### 3.1.1. Field Interview Objectives

The field interviews, which were conducted with practicing systems engineers, employed the following objectives:

1. Understand the areas of high priority attention for systems engineering automation.
2. Understand the areas and degrees of variability in systems engineering processes.

A total of 15 systems engineers were interviewed in 3 organizations. The organizations selected represented a cross-section of systems engineering of computer-based systems:

- New system development and life cycle support activities
- Government and industry
- Acquisition and in-house activities
- Small, medium and large systems

### 3.1.2. Field Interview Methods

The interviews were structured having specified questions but leaving the character of the response open (termed type II interviews [BOU79]). The questions were broad and allowed the interviewees to express themselves freely. This form was selected over a totally structured set of questions and responses (termed type I interviews [BOU79]) in order to better surmise the most important issues in the minds of the interviewees and thus more effectively address the first objective of the interviews.

Insight into the important systems engineering issues in the minds of each interviewee was obtained in two ways: 1) by allowing the maximum freedom in answers and in steering the discussions, and 2) by asking seven separate questions spaced throughout the interview for an assessment of the important issues. In the first case, the important issues can be assessed by analyzing what subjects the interviewee talked most about. In the latter case, the multiple questions provided reinforcement for what was considered most important and provided a number of opportunities to extract this information.

The interview questions were loosely structured in the following major areas:

- interviewee background profile
- application area
- organization and process
- individual roles and responsibilities
- individual activities
- individual methods
- individual automation
- interactions with other individuals and organizations
- organizational connectivity
- classified information
- method improvement
- automation improvement

The specific questions were refined over the course of the interviews. Table 3-1 shows the final version of the questions that were asked. Questions denoted with an asterisk (*) identify those that are attempting to identify those areas of greatest importance.

The questions were not strictly sequenced; the interviewers instead followed the flow of the conversation, following up with additional probing questions. This allowed the interviews to be more of a casual conversation rather than a highly structured question and answer session.

All interviewees did not address every question. Some questions that were clear from previous interviewees (e.g., what is the organizational structure) were not asked repeatedly. Questions were occasionally omitted in order to keep to the time limit. At the discretion of the interviewers, certain areas were probed in greater detail with one interviewee than others .

Table 3-1. Final Version of Interview Questions

| | |
|---|---|
| 1. Introduction | • Briefest possible explanation of why the interview is being conducted. |
| 2. Interviewee profile | • Name, organization, position, educational background, years experience |
| 3. Application area, systems | • What kind of systems are you involved in building? Describe the application domain.<br>• What are the most difficult aspects of building these kinds of systems?* |
| 4. Organization, process | • Provide an overview of the organizational structure in which you work.<br>• What sort of general process does the organization follow in building systems? How do these various organizational elements interact? |
| 5. Individual roles and responsibilities | • Describe what you do. What are your areas of responsibility?<br>• How do you fit into the organization?<br>• What do you consider the most important aspects of your job— the things that are most critical to having a successful system?* |
| 6. Individual activities | • What activities consume most of your time? What is the "pie chart" describing what you do?*<br>• What are the most difficult aspects of your job? How are they difficult?* |
| 7. Individual methods | • What sort of methods or techniques do you use in your job (with respect to the activities described above)?<br>• What information do you require or use in your job? What information do you generate?<br>• If you were training someone in your job, what would you teach him/her? What advise would you give?* |
| 8. Individual automation | • What computers do use in your job?<br>• What automated tools do you use today?<br>• How much of your work involves using the computer? |
| 9. Interactions with other individuals, organizations | • Who do you typically work with? With what other individuals or roles do you frequently interact?<br>• What type of interaction is it (voice, document, Email, fax, etc.)? |
| 10. Organization connectivity | • Do you have any electronic connectivity within your organization? with other organizations?<br>• To what extend do you use electronic mail and for what type of communication? |

Table 3-1. Final Version of Interview Questions (continued)

| | |
|---|---|
| 11. Classified information | • How much of your job involves classified information? • What type of information typically is classified and at what level? |
| 12. Method improvement | • If you could improve any aspect of your job, what would it be? * |
| 13. Automation improvement | • What automated tools or aids do you wish you had?* |

The interviews were conducted by two interviewers. Both interviewers asked questions and both took notes (in addition, the interviews were audio-taped for later reference). Tandem interviewing has been shown to have major advantages over a single interviewer, as follows: [BOU79]

- Efficiency in questioning and recording
- Increase in rapport with interviewee
- Increase in accuracy in questioning
- Increase in depth and range of data through probing and clarifications
- Increase in accuracy of analysis, with less opportunity for bias

The field interviews were conducted using the guidance of the *Handbook of Industrial and Organizational Psychology* [BOU79], as follows:

- Minimize status differentials by utilizing interviewers familiar with system engineering
- Know whom you are talking to
- Identify interviewers
- Inform all interviewee how they were selected
- Maximize privacy
- Maintain confidentiality
- Project an 'inquiring stance' to the interviewee
- Maintain respondent motivation through trust and by questioning only in areas that the interviewee knows well
- Ask broader questions before more specific or biasing ones
- Ask direct, rather than indirect questions
- Maintain neutrality
- Listen carefully
- Limit interviews to 90 minutes

Because the environment for each interviewee is substantially different, the field interviews are not equivalent to multiple samples in an experiment. Instead, each case is considered to be equivalent to a single experiment and analysis, following a cross-experiment, rather than a within-experiment approach [SWA88]. Following the interview, the data was reduced and analyzed in the following steps:

1. An individual case report was created for each interviewee that outlined the key responses for individual profiling questions (e.g., role, methods). This report organized information from the interviewers' handwritten notes and, when necessary, consulted the audio tapes.
2. An organizational profile was created that aggregated organizational issues (e.g., process, organizational structure).
3. Individual cross-case conclusions are drawn across all interviewees in an organization.
4. Organizational cross-case conclusions were drawn across all organizations.

## 3.2 Field Interview Results

This section presents the conclusions from field interviews of practicing systems engineers accomplished under the SECD program in 1990 and from previous, relevant field interviews by Microelectronics and Computer Technology Corporation (MCC) in 1986.

### 3.3.1 SECD Field Interview Conclusions

The SECD program accomplished a limited number of field interviews in 1990. The field interviews were selected to provide a cross section of systems engineering:

- Rome Laboratory C3I government in-house mission/systems requirements analysis (referred to as RL-A)
- Rome Laboratory Intelligence Data Handling Systems (IDHS) government acquisition (referred to as RL-B)
- Naval Air Development Center[1] anti-submarine warfare (ASW) avionics systems engineering (referred to as NADC)
- IBM Owego, NY, avionics systems engineering (referred to as IBM)

All interviews involved computer-based applications. The interviews provided a good cross section of government and contractor activities and a good diversity of systems engineering roles:

- RL-A and RL-B are government organizations
- IBM is a contractor organization
- NADC, while organizationally a government organization, had many characteristics similar to contractor organizations

The interviews involved systems engineering activities of various sizes:

- Small systems engineering teams (1-4 people): RL-A
- Medium systems engineering teams (5-19 people): RL-B and NADC
- Large systems engineering teams (20-50 people): IBM

The interviews spanned the life cycle:

- Early life cycle determination of mission and system requirements: RL-A
- Advanced system development: RL-B
- Production system development: IBM
- Major upgrades of fielded systems: NADC

Appendix A summarizes, in bullet form ,the composite conclusions for each of the four interview organizations. Appendix B includes the individual interview highlights. The overall SECD field interview conclusions are discussed below.

**Personnel Profile.** The systems engineers interviewed all had a minimum of a BS degree in an scientific discipline. Many had advanced engineering and management degrees. Those interviewed typically fell into one of two profiles:

- Career-long systems engineering experience, typically 20-30 years, in the application area, often starting in a specialty discipline and being promoted into a systems engineering role.
- Fast-track systems engineers with 8-10 years experience, that have rapidly demonstrated their capabilities, potentially in different application areas.

Both profiles are highly valued by their organizations for their experience and knowledge. Both demonstrate superior leadership, persuasive communications and interpersonal skills.

**Existing Automation.** The various organizations were remarkably similar in the level of existing automation for systems engineering, profiled, as follows:

- Desktop personal computer or workstation is common.
- Computing resources are networked within the organization, promoting information transfer and electronic mail.
- Electronic mail is frequently used for organizational communication and for remote communication with customers, users and contractors.
- The desktop computer provided general purpose tools for the systems engineer: word processing, project management, spreadsheet, database and drawing tools.
- General purpose databases were frequently applied for requirements tracking, action items and problem reports.
- Special-purpose automation (e.g., simulation, prototyping, analysis, requirements traceability) was often available on other computer systems.

**Systems Engineering Processes.** All organization portray systems engineering as a team activity, though there is considerable variance in the size of teams. All organizations apply processes and methods that have evolved to be custom to the particular organization, personnel and application area. All processes, in both government and contractor organizations, are largely driven by a plethora of government regulation and policies.

Many of the methods applied, reflecting current "best practices," were somewhat similar across organizations. For example, functional flow diagrams, system block diagrams, timelines, traceability matrices, and state-transition charts were commonly used. However, it is important to note that even these methods exhibited some organization-specific variances. Some methods were localized to individuals or teams.

The systems engineers' jobs, in all cases, involves access to a large amount of information. In several cases, the organization depends on the personal knowledge of *where* information can be found. Most systems engineers must deal with classified information in some capacity.

**Priority Areas.** The interviews asked many question to identify what priority areas are most important for automation. Those areas, in approximate order of priority, are listed below.

1. **Requirements definition, allocation and traceability.** Translating mission needs to systems requirements and system requirements to subsystem requirements is a difficult and complex activity.

2. **Requirements change management and impact analysis.** Requirements are constantly changing. These changes must be managed and reflected throughoui the system.
3. **Communication and collaboration.** There is a high level of interaction amongst the systems engineering team and between the team and the customer, user, and specialty engineers. Of particular priority are multi-disciplinary collaborations, team interactions, tasking and mentoring.
4. **Interface management.** Interface definition and control is a complex technical, communications and management issue that spans the life cycle.
5. **Standard and policy applicability.** An organization's systems engineering process is constrained by a myriad of government and contractor standards and policies. Finding, interpreting and applying them is a difficult and particularly annoying task.
6. **Handling of classified information.** Accessing, generating and storing classified information is typically accomplished manually using paper media. Current technologies typically preclude computer automation where classified information is involved.
7. **Tradeoffs.** Alternative analysis and trade studies are fundamental to good systems engineering. Tradeoffs typically require different engineering disciplines and involve numerous individuals.
8. **Integration and test.** The systems engineer is typically responsible for system-level integration and test. This requires continuous attention throughout the development and significant coordination in the later phases.
9. **Program management within cost and schedule constraints.** The system developments are typically faced with aggressive schedules and limited and ill-timed funds. The systems engineer faces the challenge of defining and managing a program that must both be feasible within these constraints and be technically acceptable.

### 3.3.2. MCC Field Interview Conclusions

The Design Process Group of the Microelectronics and Computer Technology Corporation (MCC), in 1986, conducted on-site interviews from 19 large software development projects to gather case studies information on actual design processes. [CUR87] While the study focused on software development projects, the team studied more general aspects of abstract requirements and design processes, organizational behavior and project communication that is applicable to the larger topic of systems engineering. Many of those interviewed were, in fact, systems engineers. The results of the research have been published in a synopsis of a number of papers here.

The interviews found that developing large software systems must be treated, at least in part, as a *learning, negotiating, and communication process*. Requirements issues were a recurring theme in the interviews: how system requirements were understood, how their instability affected design, and how they were communicated throughout the project. [KRA88]

The interviews found that the three most salient problems, in terms of the additional effort or mistakes attributed to them, were: [KRA88]

1. The thin spread of application domain knowledge
2. Fluctuating and conflicting requirements
3. Communication and coordination breakdowns

The deep application knowledge needed to successfully build most large, complex systems was thinly spread throughout the development staffs. This problem was particularly characteristic of embedded applications (e.g., avionics or telephony). The thin spread of application knowledge often manifests itself by the 'project guru' who is an exceptional designer who can map deep application knowledge into a computational architecture. This individual exerts extraordinary influence over the direction of the design team. This results in substantial efforts spent coordinating a common understanding of the both the application domain and of how the system should perform within it. Multi-company efforts often had difficulty resolving individual models of the application. [KRA88]

Requirements are as complex as the system they represent. Moreover, turning requirements into systems is an extremely dynamic task with much fluctuation. The user needs analysis behind the requirements is often a conflict-based process in which significant negotiation occurs over the entire life cycle of the system. [KRA88]

Although requirements are intended to be a stable reference for design and implementation, requirements fluctuation and conflict arises from many sources: [KRA88]

- Market or competition factors
- Customer needs and the degree of responsiveness to those needs
- Internal company sources, such as marketing, product line management
- Uncertainty due to a lack of application knowledge of the development team

- Initial assumptions that later prove to be wrong
- Design team's desire to limit requirements to meet schedule, budget and technical constraints
- Changes in the underlying technologies
- Addition of enhancements by programmers that are not required

The communication and coordination processes within a project are crucial to coping with the fluctuation and conflict amongst requirements since there is no single source of requirements or requirements changes.

Large projects require extensive communication that was not reduced by documentation. The system development involves a large number of groups to coordinate their activities, or at least share information, during development. Early phases concentrated on clarifying issues, defining terms, coordinating representational conventions and creating channels for the flow of information. Several impediments to communication were discovered. [KRA88]

- The complexity of the customer interface hindered the establishment of stable requirements
- Organizational boundaries hindered understanding the requirements
- Political barriers created a need for informal communication networks
- Temporal boundaries buried design rationale

The four types of common communication breakdowns that were found are: [KRA87]

- No communication between groups
- Miscommunication between groups
- Conflicting information from multiple sources
- Communication problems due to project dynamics, including:
  - information loss during transitions between phases
  - loss of unrecorded knowledge due to change or loss of personnel
  - exponential growth of potential communication paths as the project becomes large

These four types of communication breakdowns were found to stem from the following factors: [KRA87]

- communication skills
- existing incentive systems

- shared representation formats
- conditions of rapid change
- local jargon
- breakdowns in information capture
- cultural mores for individual behavior

Some of the potential breakdowns between groups were avoided by individuals that "span the boundary" between groups. For example, the chief systems engineer is a *boundary spanner* that translates between the needs of customers and capabilities of the designers. Often boundary spanning resulted in informal communication networks that were observed in five areas of technical information: [KRA87]

- feature and attribute negotiation
- application design
- system diagnosis
- technology awareness
- reuse

## 4. Needs Conclusions

Systems engineering activities can be categorized into three types of activities:

1. Engineering
2. Communication
3. Management

The precise mixture of engineering, communication and management activities varies by individual and will often change over the life cycle. For example, the systems engineer may begin by performing primarily an engineering role, defining requirements as part of a small team. As the project expands into system design, large amounts of the systems engineers time may be spent coordinating the activities of the subsystem engineers and disseminating a common understanding of the requirements. Once the systems design is complete and the requirements allocated, the systems engineer may be largely performing a management role, planning and tracking the subsystem developments.

A total of 18 different activities are listed below that are frequently associated with the systems engineering process:

1. **Engineering**
   - 1.1 Requirements Engineering
   - 1.2 System Design & Allocation
   - 1.3 Interface Definition & Integration
   - 1.4 Tradeoff Analysis
   - 1.5 Engineering Decision Making
   - 1.6 Change Impact Analysis & Management
   - 1.7 Integration Planning and Management
   - 1.8 Quality Engineering and Assurance
   - 1.9 Specification Generation
2. **Communication**
   - 2.1 Collaboration & Coordination
   - 2.2 Information Research
   - 2.3 Boundary Spanning
   - 2.4 Joint Work Product Development

3. **Management**
    3.1 Standard and Policy Application
    3.2 Process Management
    3.3 Program Planning & Tracking
    3.4 Task Management
    3.5 Risk Analysis

Affecting each of these activities is the problem of electronic handling of classified information. There is a need for pragmatic solutions that allow a high degree of automated support when some or all of the information involved is classified.

The following subsections overview systems engineering needs for each of these activities.

## 4.1 Engineering Needs

This section discusses the engineering needs for the following activities:

- Requirements Engineering
- System Design & Allocation
- Interface Definition & Integration
- Tradeoff Analysis
- Engineering Decision Making
- Change Impact Analysis & Management
- Integration Planning and Management
- Quality Engineering and Assurance
- Specification Generation

**Requirements Engineering.** Because the requirements are as complex as the system they represent, the tasks of organizing requirements, precisely specifying requirements without ambiguity, and insuring consistency by removing conflicting or misleading requirements, are all extremely difficult. Moreover, the process of turning requirements into systems is a very dynamic task, involving significant iteration and change.

Most programs have difficulty defining requirements and dealing with constantly changing requirements. [RED84] As much as fifty-five (55) percent of system errors are introduced during the requirements definition process. It is believed that proper (i.e., systematic, disciplined) requirements engineering would alleviate this situation.

Yet, many in-practice requirements engineering methods are being applied manually or are not adequately supported by automation. Most available automation is either general purpose tools (e.g., word processors and graphic drawing tools) or special purpose tools supporting more formalized methods (e.g., simulation or CASE tools). Many semi-formal methods that may be custom to a particular organization.

Specific requirements engineering needs include the following:

- New techniques are needed for dealing with the volume and complexity of mission, systems and subsystem requirements.
- Automated support is needed for custom, semi-formal requirements modeling techniques.
- New approaches are needed to incrementally capture, elaborate and formalize requirements.
- The requirements engineering process must be "re-engineered" to reflect it as a continuing, iterative process, rather than a discrete step in system development.
- The effort to identify, understand, propagate and respond to a requirements change must be significantly reduced.

**System Design & Allocation.** System design has many of the same needs of requirements activities because design is an extension of a large and complex system definition problem. System design defines the requirements (allocated plus derived) for the first tier segments or subsystems. Subsystem design defines the requirements for lower level subsystems or components.

The system design process is also very dynamic, involving significant iteration and change. Moreover, the design process introduces a multitude of alternatives and tradeoffs that must be considered, weighed and analyzed. In many cases, important alternatives or tradeoffs are not considered due to cost and schedule pressures. This leads to stories of design by "the seat of the pants."

Negotiation and compromise are common during the allocation process and continue, to some extent, throughout the life cycle. Change impact analysis and management becomes critical for large systems.

Specific system design and allocation needs are as follows:

- Support is needed to effectively define, evolve and view a system design from many perspectives and from the aspect of the various subsystems and specialty disciplines.
- Automated support is needed for custom, semi-formal system design techniques.
- Requirements allocation must be made more efficient and traceability must be maintained throughout the development.
- Once allocated requirements have been baselined, anticipating, controlling, tracking, and managing changes to the baseline must handled reliably and efficiently.
- Automated techniques are needed to allocate requirements and design constraints or non-functional requirements to subsystems in a manner to guarantee that they will not be violated.

**Interface Definition & Integration.** The management and support of engineering activities associated with system and subsystem interfaces is a priority concern for the system engineer. Key interface problems exist in the following:

- Identification and understanding of interface requirements
- External interface adversity (i.e., complexity of interface interactions with the surrounding environment) [DEU90]
- Proper, complete and consistent interface specification
- Early negotiation of interfaces and proper management of the interface definition throughout the life cycle
- Change propagation from interfaces and across interfaces

Specific interface definition and integration needs are as follows:

- Interface requirements and definitions need to be controlled from the earliest identification and partitioning of the system.
- Automated support is needed for better managing the many complex interrelationships that exist across interface boundaries.
- Support is needed for managing and negotiating interfaces amongst many participants.
- Automated change impact analysis is needed to assist in assessing the impacts of interface changes and propagating the affects to the appropriate subsystems.

- Better specification and design representations are needed for complete interface specifications.
- Automated techniques are needed for guaranteeing the consistency of subsystem specifications and designs to baseline interface definitions.

**Tradeoff Analysis.** The major problem with tradeoff analyses is the cost, time and effort to accomplish them. A tradeoff analysis can represent a major diversion in the process simply to "answer a single question." Poor decisions are often made because of the lack of proper analysis or because sufficient alternatives were not thoroughly analyzed.

Typically, once a tradeoff analysis is accomplished, only the resulting decision is remembered. If the issue needs to revisited, the previous analysis is not available and the rationale for the decision is lost. As a result, even good analysis may be rendered useless if later factors change the decision arbitrarily.

Specific tradeoff analysis needs are as follows:

- Tradeoffs and decision rationale need to be captured and accessible for later review or reconsideration.
- The tradeoff process needs to be more productive in order to stimulate more thorough analyses.
- Automated support for tradeoffs need to support multi-disciplinary group processes that occur in a concurrent engineering approach.
- Integration mechanisms are needed to better support the use of multiple analysis tools and the synthesis of their results.
- Traceability needs to maintained between tradeoff decisions and the areas that they impact.

**Engineering Decision Making.** There are countless decisions made throughout a system development at all levels. Today, the decision making process primarily relies on human actions and interactions that often fail or insufficient in large organizations. Specific needs in this area include the following:

- Each decision needs to be specifically identified with all of the areas that the decision impacts.
- There is a need to reliably disseminate the decision to all those affected to direct a correct, consistent and unified response to the decision.
- The interrelationships between decisions need to be better understood.

- The resulting changes and actions from a decision need to be tracked to insure that the decision is completely and consistently reflected throughout the system development.
- Recording the decision and supporting rationale for decisions is needed to enable the inevitable revisiting or retraction of decisions.
- Utilities are needed to search and browse the decision history of a project.

**Change Impact Analysis & Management.** Effective change impact analysis is area of great need that would have significant benefit across the life cycle. Currently, simple traceability matrices are the systems engineer's only impact analysis tools. Effective change impact analysis goes far beyond simple traceability. Needs in this area include the following:

- Traceability needs be automatically captured and maintained as part of the specification, design and allocation processes.
- There is a need for more sophisticated and thorough means for identifying exactly what is impacted as a result of a proposed change.
- More reliable mechanisms are required for assessing, in terms of cost, resources and schedule, the program impact of changes.
- Support is needed for managing the change process across many individuals, in different organizations, and across various subsystems and work products.

**Integration Planning and Management.** Early in the life cycle, the systems engineer is involved in integration planning and in interface definition and management. Integration builds must be planned consistent with individual subsystem development and supplier schedules. Interfaces must be completely and consistently negotiated and specified.

Later in the life cycle, the systems engineer is faced with two significant challenges:

- Maintaining the consistency of the interface definitions in response to numerous changes
- Assessing the compliance of subsystems or suppliers in adhering to the interface specification and not violating any other total system design constraints

The systems engineer attempts to anticipate and respond to integration problems before they occur. Once integration has begun, the systems engineer is in a tracking and reactive mode, attempting to minimize the impacts of integration problems on the overall system cost, schedule, or mission.

Specific needs in this area include the following:

- Better mechanisms are needed to couple the integration planning activities with the design process.
- Automated support is needed to maintain the consistency of integration plans as the design changes.
- Integration impacts need to identified from other system baseline changes.
- Automated techniques are needed for guaranteeing the consistency of subsystem specifications and designs to baseline interface definitions, even if those design specifications are controlled by remote subcontractors or suppliers.
- Supplier management needs to be better integrated into the development process.
- Contingency management is needed to better insure that integration schedules are met.

**Quality Engineering and Assurance.** Quality assurance, imposed from the start and throughout system development efforts, is needed to reduce the risk of failure. [AFS89] There is a great need for the automated support of "total system quality" in systems engineering. Assurance is becoming an increasingly more difficult task for large and complex systems, while at the same time becoming more essential for safety, security, and trustworthiness.

Specific needs in this area include:

- Quality needs to be driven by and tailored to specific project needs.
- More efficient ways are needed to accomplish incremental review and critique of information.
- Automated metrics support is needed for the measuring key quality aspects of work products and the process.
- Better ways are needed to view, compare and analyze information or metrics data from different sources.
- Automated techniques are needed to focus human attention on only those problems areas or areas of interest.
- Statistical quality assurance techniques need to integrated throughout the development process.
- Automated support is required to alert those individuals necessary to take action of critical events, metrics or observations.

**Specification Generation.** Documentation often accounts for 20-30% of the cost of a large system development. Yet studies have shown that MIL-SPEC documentation is not an effective communication mechanism.[KRA88] Automated methods are needed to significantly reduce the unnecessary cost of specification development and production. Specific needs in this area are as follows:

- Support is needed for better understanding and tailoring of data item descriptions and requirements.
- Better automated support is needed for describing and maintaining document templates and contents to automated documentation tools.
- Bi-directional automated support is needed for reflecting changes in edited documents and the sources for the information.
- Better forms and representations are needed for communicating information for review and analysis.

## 4.2 Communication Needs

This section discusses the communication needs for the following activities:

- Collaboration & Coordination
- Information Research
- Boundary Spanning
- Joint Work Product Development

**Collaboration & Coordination.** The system development involves a large number of groups to coordinate their activities, or at least share information, during development. Early phases concentrated on clarifying issues, defining terms, coordinating representational conventions and creating channels for the flow of information. Several impediments to communication were discovered. [KRA88]

- The complexity of the customer interface hindered the establishment of stable requirements
- Organizational boundaries hindered understanding the requirements
- Political barriers created a need for informal communication networks
- Temporal boundaries buried design rationale

The four types of common communication breakdowns that were found are: [KRA87]

- No communication between groups

- Miscommunication between groups
- Conflicting information from multiple sources
- Communication problems due to project dynamics, including:
  - information loss during transitions between phases
  - loss of unrecorded knowledge due to change or loss of personnel
  - exponential growth of potential communication paths as the project becomes large

Specific collaboration and communication needs are as follows:

- Broader automated support is needed to support multi-media, group communications.
- Information capture and recall facilities need to be better integrated into electronic communication mechanisms.
- Communication utilities needs to better support formal communication process requirements.
- The information sharing process needs to include the automatic notification of selected individuals of new, important information.
- Utilities are required to better organize, prioritize, manage volumes of electronic mail.

**Information Research.** The systems engineer utilizes a broad spectrum of information: reference information, information particular to a program, and information about previous programs. The information research needs can be characterized as having three parts:

- Capturing or providing electronic access to a wide variety of heterogeneous, multi-media information
- Providing facilities for effective search of specific information from the resulting vast store of information
- Easy retrieval and application or reuse of information assets

Information research needs include the following:

- The vast amount of information that is generated during a system development needs to be automatically captured, indexed, and organized.
- Application and system knowledge needs to be readily available to the various members of the system development team.

- Meaningful classification information (e.g., keywords) needs to be automatically extracted from reference information.
- New user interface paradigms are needed to promote searching and browsing large, complex heterogeneous information stores.
- Automated assistance is required by users in formulating, executing and refining search strategies.
- Better approaches are needed to extract, adapt and reuse information fragments.

**Boundary Spanning.** Current automation, at best, supports information sharing. New approaches are needed for supporting and facilitating group interactions electronically and supporting group processes. Specific needs include the following:

- Practical electronic substitutes for meetings are needed to lessen the burden of group coordination.
- Support is needed to reconcile different jargon and terminologies that occur when multiple disciplinary interaction occurs.
- New user interface paradigms are needed for collaborative group processes such as brainstorming and negotiation.
- Standards are needed to support "groupware" across heterogeneous computing platforms.
- Innovative ways for detecting and responding to communication and coordination breakdowns are needed.

**Joint Work Product Development.** Current document product automation does not explicitly recognize that many work products are group produced. Automation needs to specifically address group work product development, as follows:

- Support is needed for convenient work product partitioning, allocation and assignment.
- Information editing facilities need to support controlled simultaneous editing of work products.
- Automation is required for the work product review, markup and markup synthesis process.
- Support is needed for coordination and tracking of work product contributions.
- Automated support is required to synthesize work products from individual contributions.

## 4.3 Management Needs

This section discusses the management needs for the following activities:

- Standard and Policy Application
- Process Management
- Program Planning & Tracking
- Task Management
- Risk Analysis

**Standard and Policy Application.** In practice, many contractual guidelines and/or details are not carried out, or adhered to. These guidelines are generally specified in great detail for good reasons, yet very often are not implemented (i.e., lost in the shuffle), to the detriment of the project. Deciding what standards and DIDs to include on a contract is a difficult and confusing task. Needs in this area include the following:

- The various standards, DIDs, policies and guidelines that apply to project need to be available on-line for query and browsing.
- Support is needed for process modeling and tailoring that is compliant with the various standards and policies.
- Expert assistance is needed in the standards tailoring and synthesis process.
- Automated means are needed to guarantee continued compliance with important standard and policy provisions throughout the system development.

**Process Management.** There is a need for innovative acquisition *tailoring* and alternative development processes such as incremental development and prototyping. For example, "user involvement should be tailored for each program, varying from cases requiring very limited involvement to ones in which a user will assume the lead role." From user involvement to development process models, it is acknowledged that no single acquisition strategy can possibly serve all situations. [AFS89] Process management needs include the following:

- Automated support is needed for process creation, assignment, execution, enforcement, and tracking.
- Support is needed for preparing standard or prescribed process templates and their application and tailoring.

- A variety of process guidance and enforcement approaches are needed to deal with different situations.
- Process automation needs to support informal, as well as formal, processes.
- Process management needs be be integrated with program management automation.

**Program Planning & Tracking.** Systems engineers will inevitably do some level of planning, scheduling, estimating and tracking, although the amount of planning and tracking activities will vary by individual and by organization.

Key problems in project planning and tracking include:

- Accurate and reliable estimation
- Scheduling within constraints
- Staff planning
- Interface to various reporting systems
- Managing plan changes

Key needs in this area include the following:

- Program management automation needs to be more flexible to support plan manipulation in any of the various plan representations.
- Support is needed to reconcile plans at various levels and to maintain consistency between high level and low level plans.
- Program management needs to better support contingency planning and management.
- Making changes to plans and executing those changes needs to be streamlined.
- Automated support is needed to assist in generating and evaluating alternative plans within defined constraints.
- Support is needed to detect interrelationships and conflicts in separate organizational and program plans.
- Automated support is needed to translate between different institutionalized program management and accounting systems.

**Task Management.** Because of the multi-disciplinary nature of the concurrent engineering process, careful tasking and monitoring of the activities of a may individuals must be accomplished. Key problems in tasking include:

- Conveying and enforcing the proper process for the task
- Maintaining proper priorities
- Effective time management
- Accomplishing optimal staffing and tasking of the staff
- Monitoring and tracking progress

Task management needs include the following:

- Support is needed to quickly define and assign new tasks in a group collaboration environment.
- Task monitoring and notification support is required.
- Individuals need automated aids for task multiplexing and dynamic prioritization.
- Group scheduling support is required, particularly in identifying available meeting times.
- Dynamic task delegation and partitioning is required.

**Risk Analysis.** The proper management of risks is critical to the success of systems engineering. Unprecedented systems—"systems for which there has been no precedent in the form of similar systems or systems performing the same functions, or for which the design teams lack full or applicable system design experience"—should receive special attention with regards to risk reduction. [AFS89]

Efforts should be made to identify and control risks as early as possible in systems development. For example, policies should mandate the use of risk management plans. Proper risk-driven analysis of alternatives may at any time redirect the development effort in terms of rework, step sub-setting, or specifying/adjusting design-to-cost levels-of-effort.

Specific needs in this area include:

- Automated support for risk management is needed that includes the identification, analysis and tracking of potential risks.
- Risk analysis needs to be integrated with contingency planning.
- Metrics and monitors needs to be supported to automatically detect risky conditions.
- Alternative plans and contingencies need to be quickly retrieved and executed when required.

# Appendix A Interview Composites

## Rome Laboratory -A Composite Profile

### Composite profile of organization

1. **Application area, systems**
   - ***Application domain***
     - C3I advanced development systems
     - small, one-of-a-kind systems, 1-5 people
2. **Organization, process**
   - ***Organizational structure***
     - Advanced Concepts
     - early life cycle involvement
     - apart from large number of external organizations
       - **technologies**
       - **threats**
       - **mission requirements**
       - **operations**
     - "nobody here a true systems engineer"
   - ***Systems engineering process***
     - Understand and refine threat scenario
     - Understand and refine mission requirements
       - **Statement of Needs often takes years to finalize**
     - Derive system requirements
       - **top-down analysis**
       - **System Operations Concept Document**
     - Simulate the system
     - Prototype the system (new)
     - Perform design tradeoffs
     - Decompose system into subsystems
3. **Automation**
   - ***Computers***
     - some computer use
     - PCs are available

- *Automated tools for PC*
  - project management
  - spreadsheet
- *Larger system tools*
  - PSL/PSA requirements traceability
  - prototypes
  - simulations

4. **Organization automation and connectivity**
   - *Electronic connectivity*
     - limited use
5. **Other general observations**

**Overall conclusions**

**Interviewee profile**

- *systems engineering background*
  - large amount of experience (20-30 years)
- *value to the organization*
  - broad base of experience and knowledge

**Priority areas**

- *translating mission requirements into system requirements*
  - understanding threats and mission needs
  - translating into a good set of mission requirements
- *tradeoffs*
  - design tradeoffs
  - simulations
  - prototypes

**Individual activities**

- *data flow for operator functions*
- *N2 chart*
- *system decomposition*
- *threat/mission/system simulation*
- *prototyping*
- *program planning*

**Methods applied**

**Interactions**

- ***level of interaction***
  - high degree of interaction
- ***high frequency interactions***
  - external organziations
  - other systems engineers
  - subsystem engineers
- ***team-orientation***
  - small teams, 1-5 people
- ***types of interaction***
  - phone. personal, meetings

**Automation**

- ***use of computer***
  - limited

**Classified Information**

- ***amount of classified***
  - large amount of classified information

## Rome Laboratory -B Composite Profile

**Composite profile of organization**

1. **Application area, systems**
   - *Application domain*
     - intelligence data handling
   - *Characteristics*
     - lots of interfaces
     - distributed user base
2. **Organization, process**
   - *Organizational structure*
     - IDHS group
     - lab - apart from, and coordinating with, user organizations
     - acquisition manager for contractor developing a segment
   - *Systems engineering process*
     - validate requirements and produce assessment report

- acquire system or upgrade
    - **perform acquisition functions to get underway**
    - **work closely with contractor during development**
        - ***manage and control requirements changes***
        - ***allocate requirements***
        - ***review design***
        - ***address problems/questions/issues as team***
    - **control interfaces**
        - ***manage ICDs***
        - ***early testing of interfaces***
        - ***coordinate with various segments***
    - **participate in reviews**
    - **maintain requirements traceabililty to requirements database**
    - **work with users on requirements, testing and transition**
- government test director
    - **write test and evaluation master plan**
    - **test readiness and documentation review**
    - **hardware configuration verification reviews**
    - **CSCI integration**
    - **systems test**
    - **alpha test (at RL)**
    - **beta site (at user sites)**
    - **site system acceptance test**
    - **OT&E with lead test site**

3. **Automation**
    - *Computers*
        - PCs generally available
        - portable Mac
        - LONEX facilities
    - *Automated tools on PC*
        - E-mail
        - word processors
        - file compare

- CM tool
- ***Mac tools***
  - Filemaker
  - Oracle
  - Excel
  - Powerpoint
  - MacDraw
- ***Other tools***
  - requirements database on LONEX facilities

4. **Organization automation and connectivity**
   - ***Electronic connectivity***
     - within organization and with contractor
5. **Other general observations**

**Overall conclusions**

**Interviewee profile**

- ***systems engineering background***
  - BS minimum
  - 15-20 years experience
- ***value to the organization***

**Priority areas**

- ***requirements***
  - incomplete, inappropriate requirements
  - changes throughout, through maintenance
  - translating user requirements into system requirements and allocating
  - requirements traceability and control
- ***communications and team interactions***
  - team interactions
    - **with contractor**
    - **with users**
  - reviews, walkthroughs
  - commenting on work products
  - keeping track of tasks, actions
- ***integration and testing***

    - interface control
    - test planning
    - testable requirements
    - test execution and coordination
- ***classified information***
    - transferring
    - managing
- ***applicability of standards***
    - understand, select, tailor

**Individual activities**

- ***traceability***
- ***close control of actions, problems, tasks***
- ***high degree of interaction***
    - users
    - contractor
- ***documentation***

**Methods applied**

- ***high degree of use of databases***

**Interactions**

- ***level of interaction***
    - very high
- ***high frequency interactions***
    - contractor
    - user
- ***team-orientation***
    - part of contractor team
    - team involvement with users
- ***types of interaction***
    - meetings, reviews
    - fax, E-mail
    - documents
    - databases

**Automation**

- ***use of computer***

        - moderate

**Classified Information**

- ***amount of classified***
    - large amount
- ***type of classified information***
    - mission requirements
    - test data
    - ICD
    - data dictionary
    - maybe SSS

## NADC Composite Profile

**Composite profile of organization**

1. **Application area, systems**
    - ***Application domain***
        - airbourne anti-submarine warfare avionics
        - life cycle support activity - performing system evolution via upgrades
        - 100s of manyears involved
    - ***Trends***
        - system getting increasingly complex
        - system engineering has greater reliance on subsystem engineering using a concurrent approach
2. **Organization, process**
    - ***Organizational structure***
        - NAVAIR PMA is the customer
        - NADC is the technical agent for NAVAIR
            - **Systems engineering departments**
                - ***systems engineers perform senior technical activities***
                - ***project engineers perform programmatic activities***
            - **Technology (specialty/subsystem engineering) depts**
                - ***radar***

        - ***acoustics***
        - ***software***
        - ***sensor***
    - **Analysis departments**
        - ***warfare analysis***
        - ***reliability***
        - ***human factors***
        - ***testing***
        - ***training***
- *Systems engineering process*
  - NAVAIR (customer) defines mission requirements
    - **operational requirement (OR) or tentative operational requirement (TOR)**
      - ***1 page from Fleet***
    - **development option paper (DOP)**
      - ***25 pages addressing OR or TOR***
      - ***how it would be integrated into the platform***
    - ***5 options with cost and recommendations***
    - **technical approach**
  - Systems engineering may support NAVAIR in the development of option papers or technical approach
  - Systems engineering supports development of ECPs
  - Systems engineering takes mission requirements to system requirements
    - **system specification**
    - **interface definitions**
    - **test plan/req's/procedures**
  - System/subsystem is acquired/developed
  - Systems engineering performs a monitoring function during development
  - Systems engineering gets involved again during testing
    - **monitors contractor development testing**
    - **supports directly tech eval for PMA, independent of contractor**

            - **not involved in op eval accomplished by independent Navy organization**
        - Fix operational problems
        - Production
        - (Basic process is stable)

3. **Automation**
    - ***Computers***
        - PC 286 on desk
        - avionic test and integration labs
        - new Suns being purchased for system engineering
    - ***Automated tools for PC***
        - word processor (Word)
        - spreadsheet (Lotus)
        - database (Dbase)
        - graphics
        - project management (Harvard Project Manager)
4. **Organization automation and connectivity**
    - ***Electronic connectivity***
        - networking within NADC
        - some E-mail links to other sites (NAVAIR is coming)
        - Email only can be used for informal interactions
5. **Other general observations**

**Overall conclusions**

**Interviewee profile**

- ***systems engineering background***
    - basic technical degree
    - 20-30+ years experience
    - moved from functional or specialty area to a systems engineer
- ***project engineer background***
    - basic technical degree
    - varied experience
        - **up and coming young engineers (5-7 yrs)**
        - **one at end of career (31 years)**
    - moved from functional area into project engineer

- ***value to the organization***
  - lots of application experience
    - **most of career in the same application**
    - **hands-on, bottom-up experience**
    - **mentors in the organization**
  - attitude
    - **eclectic, generalist**
    - **open-minded interest, ask questions**
    - **assertive, leadership**
    - **do whatever is needed**

**Priority areas**

- *Requirements engineering*
  - taking mission requirement to system requirements
    - **within constraints**
    - **dealing with complexity**
  - requirements decomposition and allocation
  - requirements tracking throughout the life cycle
  - requirements change
- *Collaboration/coordination*
  - collaborative process of multi-functional team for requirements development
  - systems engineering role of the moderator, glue
  - mentoring, answering questions
- *Scheduling*
  - scheduling within constraints
  - maintaining proper priorities
  - interface to various reporting systems
  - effective time management
- *Interface management*
- *Standard and policy applicability*
  - deciding what standards and DIDs to include in contracts

**Individual activities**

- ***note that this was stated as an open-ended question; differences in viewpoint and terminology make these exact percentages questionable***
- ***"pie chart"***
  - always frequent activities
    - **coordination (min 20%, max 90%)**
    - **documentation(10-25%)**
  - sometimes frequent activities - depending on their job
    - **planning**
    - **analysis**
    - **engineering**

Methods applied

- ***note that most could not answer directly the question on what methods were applied***
- ***early life cycle***
  - block diagrams
  - timeline diagrams
  - hierarchy of functions
  - functional flow diagrams
  - system decomposition
  - alternatives analysis
  - trade studies
  - specific analyses
  - models
  - critical parameter tracking
- ***information***
  - vast amount of information used, generated, reviewed
  - generally knows where to look for it
    - **finding it is not a problem for those who know**
    - **other ask the more experienced when they cannot find it**

Interactions

- ***level of interaction***
  - high level of interaction
  - 50% external, 50% internal

- ***high frequency interactions***
  - project engineer - systems engineer
  - systems/project engineer - customer
  - systems/project engineer - technology/analysis support
  - mentoring - answering questions
- ***team-orientation***
  - close involvement of technology and analysis personnel in a concurrent approach
  - team-oriented interactions with little conflict
  - two uses of multi-functional team development of system spec
- ***types of interaction***
  - voice and paper interactions most frequent
  - frequent use of fax
  - use of Email for informal communication is increasing

**Automation**

- ***use of computer***
  - some used rarely for the following reasons
    - **interaction-intensive job**
    - **classified information**
    - **not very computer literate**
  - *half were frequent users*

**Classified Information**

- ***amount of classified***
  - some frequent, all had some
- ***type of classified information***
  - mission information is typically Secret
  - system requirements info is typically Confidential
  - development information is typically Unclassified

## IBM Owego Composite Profile

**Composite profile of organization**

1. **Application area, systems**
    - ***Application domain***
        - avionics
        - electronics support
        - C3I
        - avionics processors
        - computer integrated manufacturing
    - ***Characteristics***
        - large systems ~5000 requirements
        - large programs - up to 1000 people may touch a program
2. **Organization, process**
    - ***Organizational structure***
        - matrix organization
            - **Program Management**
            - **Avionics Systems - systems engineers**
            - **Software**
            - **IV&V**
            - **I&T**
            - **HW**
            - **speciality disciplines (quality, reliability, logistics)**
    - ***Systems engineering process***
        - Pre-RFP
            - **talking to the customer**
            - **refine operational need**
            - **do mission analysis**
            - **perform market research**
        - RFP-Proposal
            - **appoint a lead engineer**
            - **put requirements into requirements management system**
            - **allocate each requirement to "owners"**

        - **clarify the requirements**
        - **identify technical performance measures**
        - **identify and rate (L,M,H) risks**
        - **do Technical Management Plan**
        - **establish WBS**
        - **setup baseline control procedures**
        - **form and chair Engineering Integration Working Group**
        - **do Systems Spec, Segment Specs, mission analysis, maybe early trade studies, some System Design Document**
    - Program execution
        - **conduct proposal requirements review**
        - **understand operational need**
        - **define/refine high level system requirements (PIDS)**
        - **write master test plan**
        - **allocate requirements to subsystems**
        - **do design analysis report**
            - ***algorithms***
            - ***moding***
            - ***higher level functional design***
        - **write subcontract hardware specs**
        - **write lower level requirements specifications (e.g., SRS)**
        - **write design description document**
        - **do interface control documents**
        - **work with subsystem designer during subsystem design**
        - **write integration and test plans**
        - **do test procedures**
        - **participate in integration and test**
        - **conduct system test**
        - **conduct flight test**

3. **Automation**
    - ***Computers***
        - PC AT or PS/2 on desk
        - Mainframe (VM) for databases
    - ***Automated tools on PC***

        - editors
        - Bookmaster - documentation, charts
        - Markup tool
        - Drawmaster graphics tool - block diagrams, flowcharts
        - PERT
        - RMAT - scans and parses out wills and shalls
    - ***Automated tools on mainframe***
        - PSL/PSA
        - requirements management system
    - ***Special purpose tools***
        - simulators
        - virtual avionics prototyping
        - ABE - Teknowledge queuing model
        - RESQUE queing model
        - APL tools

4. **Organization automation and connectivity**
    - ***Electronic connectivity***
        - organization-wide network
        - PROFS (e-mail, file transfer, calendar, meeting mgmt)
5. **Other general observations**
    - ***high level of automation***
        - requirements database
        - multiple tools employed
    - ***high level of standard processes, guides, training***

**Overall conclusions**

**Interviewee profile**

- ***systems engineering background***
    - BS is a minimum, several with advanced degrees
    - 8-10+ years experience
        - **minimum 5-8 years to become systems engineer**
- ***value to the organization***
    - experience

    - communication

**Priority areas**

- ***requirements***
    - defining
    - allocating
    - "keeping it all together"
    - assessing compliance
    - change management
    - good requirements spec for vendors
- ***communication***
    - interfacing with people
    - multi-disciplinary communication
    - vendor communication
- ***cost and schedule constraints***

**Individual activities**

- ***communication and interaction***
- ***requirements, design, test analysis and derivation***
- ***work product development***

**Methods applied**

- ***allocation, traceability***
- ***multi-view requirements analysis***
    - functional - PSA templates
    - physical - system block diagrams
    - operational - homegrown methods
    - three views manually associated - no central database
- ***reuse***
    - proposals
    - specifications
    - tests

**Interactions**

- ***level of interaction***
    - communication occupied the majority of day-to-day activities
- ***high frequency interactions***
    - systems engineering team

    - subsystem, specialty engineers
    - vendors
    - customers
  - ***multi-disciplinary systems engineering team***
    - 12-15 systems engineers
    - 10+ programmers
    - 6+ hardware engineers
    - quality, reliability, logistics, training
    - weekly meetings
  - ***types of interaction***
    - informal - voice, phone, meetings, E-mail
    - formal - specs, databases, documents, directives

**Automation**

  - ***use of computer***
    - high degree of computer usage
    - requirements database requires continuing use
    - electronic markup of documents

**Classified Information**

  - ***amount of classified***
    - none discussed

## Appendix B. Rome Laboratories Interviews (Series A)

**Interview questions 1.**

1. **Interviewee profile**
    - *Name, organization, position, educational background, years experience*
        - BS Math, Physics
        - 30 years experience
2. **Application area, systems**
    - *What are the most difficults aspect of building these kinds of systems?*
        - knowing the threat, being able to detect, track and respond
3. **Individual roles and responsibilities**
    - *Describe what you do. What are your areas of responsibility?*
        - translate threat scenarios to mission requirements to system requirements
    - *How do you fit into the organization?*
        - Advanced Concepts
    - *What information do you require or use in your job?*
        - large volume of information on:
            - intelligence
            - communications
            - multi-level security
            - surveillance
            - intelligence fusion
            - radar
            - digital messages
            - sensors
            - information presentation for decisionmaking
            - capabilities of our forces
            - command and control
            - weapons systems
        - threat scenarios
            - reports, text, graphics

    - Statement of Needs
    - System Operations Requirements Document

4. **Individual activities**
   - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
     - Understand and refine threat scenario
       - **threat is "approved" by an other organization**
       - **threat scenarios come from yet another organization**
       - **develop technical intelligence on tyreat**
       - **Postulate how enemy would use**
       - **develop scenarios**
     - Understand and refine mission requirements
       - **Statement of Needs often takes years to finalize**
       - **performs operations analysis**
         - ***understand what the operators do***
         - ***look at previous systems***
     - Derive system requirements
       - **System Operations Concept Document**
     - Simulate the system
       - **establish database: parameters and attributes of threats**
       - **develop threat scenarios**
       - **Develop database of different systems/subsystems**
       - **"steal" models of communications, sensors**
       - **Use simulation to derive ops requirements and to evaluate proposed systems**
     - Decompose system into subsystems
       - **coordinate with various subsystems**
       - **perform interface definition and control**
   - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
     - translating mission requirements to system requirements
   - ***What makes you successful, good, valuable to the organization?***
     - broad base of knowledge, lots of experience

- ***What are the most difficult aspects of your job? How are they difficult?***

5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
     - data flow for operator functions
     - N2 chart
     - system decomposition
     - threat/mission/system simulation
   - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
     - mentoring: need actual experience of doing it (5-10 years)
6. **Individual automation**
   - ***What computers do use in your job? What others are available to you?***
   - ***What automated tools do you use today?***
     - report generator
     - PSL/PSA
     - Network 2.5
   - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
     - other systems engineers
     - subsystems engineers
       - pyramid of systems of and subsystems
       - subsystems may not be under your control
   - ***What interaction do you have with other roles or organizations?***
     - many external organizations
       - for systems need info
       - for intelligence info
       - for ops info
   - ***What type of interaction is it (voice, documents, etc.)?***
     - documents, one-on-one, meetings

- *How much of your time is consumed with this interaction?*

8. **Organization automation and connectivity**
   - *What tools do the other organizations that you work with use? On what computers?*
   - *Do you have any electronic connectivity with these systems?*
     - none evident
9. **Method improvement**
   - *If you could improve any aspect of your job, what would it be?*
10. **Automation improvement**
    - *What automated tools or aids do you wish you had?*
11. **Classified Information**
    - *How much of your job involves classified information?*
      - much of information is classified
12. **Other comments/observations**

**Interview questions 2.**

1. **Interviewee profile**
   - *Name, organization, position, educational background, years experience*
     - graduate degree Systems Management
       - **Defense Systems Management College**
       - **certified level 4 acquisition manager**
     - 21 years in Air Force acquisition, 6 years at RL
2. **Application area, systems**
   - *What are the most difficults aspect of building these kinds of systems?*
     - funding: timing of funding, funding uncertainty
     - lead times in the procurement process
       - **change orders can take 6-12 months**
3. **Individual roles and responsibilities**
   - *Describe what you do. What are your areas of responsibility?*
     - acquisition management
       - **laying out a roadmap**

        - **defining funding, programmatic aspects**
- ***How do you fit into the organization?***
    - Advanced Concepts
    - "nobody here a true systems engineer"
- ***What information do you require or use in your job?***
    - threat/mission scenarios
    - system requirements
    - system design
    - design tradeoffs

4. **Individual activities**
    - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
        - "job changes substantially depending on the phase"
        - acquisition management is primarily current job
            - **laying out a roadmap**
            - **defining funding, programmatic aspects**
        - requirements identification
            - **top-down, agonizing**
        - system decomposition
        - rapid prototyping
            - **starting to play with prototyping**
        - design tradeoffs
            - **think about the whole system**
            - **across all disciplines**
        - test phase
            - **requirements verification**
    - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
        - "good engineering is the tradeoff process"
    - ***What makes you successful, good, valuable to the organization?***
    - ***What are the most difficult aspects of your job? How are they difficult?***

5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
     - prototyping
   - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
     - mentoring via case studies
       - **"figuring out which are big alligators and little alligators"**
6. **Individual automation**
   - ***What computers do use in your job? What others are available to you?***
   - ***What automated tools do you use today?***
     - requirements tools for traceability
     - PC-based project management system
     - spreadsheet
   - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
     - high degree of interaction
   - ***What interaction do you have with other roles or organizations?***
   - ***What type of interaction is it (voice, documents, etc.)?***
   - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
   - ***What tools do the other organizations that you work with use? On what computers?***
   - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
   - ***If you could improve any aspect of your job, what would it be?***
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
      - more timely CSCS/C
11. **Classified Information**
    - ***How much of your job involves classified information?***
12. **Other comments/observations**

## Appendix C. Rome Laboratories Interviews (Series B)

**Interview questions 1.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - BS EE
        - working on Masters System Development & Management
        - 14 Years experience
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
        - involvement with users
            - **requirements volatility**
            - **incomplete requirements**
            - **"defining requirements in terms of what they know, not what they need"**
        - development standards
            - **understand what they are**
            - **level of detail**
            - **working out agreements**
            - **e.g., 238 data items on a contract, 40-50 DIDS**
        - testing
            - **"... is a nightmare"**
            - **can't exhaustively test**
            - **how much is enough**
            - **resolve during OT&E**
        - maintenance
            - **problems**
            - **requirements changes**
3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
        - project engineer
    - ***How do you fit into the organization?***

    - IDHS organization
- ***What information do you require or use in your job?***
    - assessment report
    - 2167 documents
    - training plan
    - installation plan
    - transition plan
    - PDR, CDR packages
    - TEMP
    - Document review report (DRR)
    - design problem report (DPR)
    - ICD
    - test data
    - data dictionary

4. **Individual activities**
    - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
        - validate requirements
            - **develop assessment report**
                - ***impact***
                - ***cost (development)***
                - ***design approach***
                - ***schedules***
                - ***importance of requirement***
                - ***risks***
        - Acquire system or upgrade
            - **assessment report becomes work plan**
            - **SOW or mod sent to contractor**
            - **mini-proposal received & reviewed**
            - **perform cost assessment**
                - ***accuracy based upon experience***
            - **understand priority of requirement**
        - Work with contractor during development
            - **close working relationship with contractor**

        - **tailor DIDS as appropriate**
            - ***work with contractor on what is captured and what is not***
            - ***notes capture these agreements***
        - **review requirements allocation**
        - **review design**
        - **address requirements changes**
            - ***assess changes between proposal & contract award***
            - ***in design - because involvement with users is crystalizing the requirements***
            - ***derived requirements***
        - **PDR**
    - *maintain requirements traceability*
        - **reqs database maintained across systems by another organization**
        - **RL maintains traceability, testability**
    - work with users
        - **Pre-PDR**
        - **2167 documents**
        - **training plan**
        - **installation plan**
        - **transition plan**
        - **requirements traceability**
    - government test director
        - **write test and evaluation master plan**
        - **CSCI integration**
            - ***contractor test plans***
        - **systems test**
            - ***contractor test plans***
        - **alpha test (at RL)**
            - ***government test plans***
        - **beta site (at user sites)**
        - **site system acceptance test**
        - **OT&E with lead test site**

        - Percentages
            - **10% systems interfacing**
            - **25% site coordination**
            - **10% AFEA coordination**
            - **33% contractor coordination**
            - **rest: personal training, working, consulting**
    - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
    - ***What makes you successful, good, valuable to the organization?***
    - ***What are the most difficult aspects of your job? How are they difficult?***
        - communications
            - **user**
            - **contractor**
            - **RL**
            - **"most of the messes..."**
        - transferring/dealing with classified information

5. **Individual methods**
    - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
        - traceability
    - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
        - give time to train to maintain currency
        - participate in working study group
        - consult with other programs
6. **Individual automation**
    - ***What computers do use in your job? What others are available to you?***
        - PCs in office
        - Mac portable
        - LONEX resources
    - ***What automated tools do you use today?***
        - requirements database

- E-mail
- scheduling tools (CPM)
- DRR, DPR forms database on Macs

- *How much of your work involves using the computer?*

7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
     - contractor
       - **100 days on site**
       - **daily meetings with management**
       - **reviews and working meetings**
       - **review design documentation**
       - **find out about a lot of things early**
     - users
   - ***What interaction do you have with other roles or organizations?***
   - ***What type of interaction is it (voice, documents, etc.)?***
     - on-site personal interactions, meetings review
     - review of documents
     - review/problem reports
     - electronic mail
       - **status**
       - **answer questions**
       - **keep in touch**
   - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
   - ***What tools do the other organizations that you work with use? On what computers?***
   - ***Do you have any electronic connectivity with these systems?***
     - with requirements database
     - E-mail with contractor
9. **Method improvement**
   - ***If you could improve any aspect of your job, what would it be?***
     - need more guides, checklists

        - **"we have a lot of young engineers, losing expoerience base"**
    - deal with applicability of other standards

10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
        - documentation control systems
            - **including configuration control**
11. **Classified Information**
    - ***How much of your job involves classified information?***
        - significant degree
12. **Other comments/observations**

**Interview questions 2.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - AS EE, BS EE, MS systems engineering
        - 8 yrs at RL, 2 years prior with contractor
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
    - ***How do you fit into the organization?***
    - ***What information do you require or use in your job?***
        - mission requirements
        - system requirements
        - ICDs
        - DRR
        - 2167 documents
4. **Individual activities**
    - ***What activities consume most of your time? What is the "pie chart" describing what you do?***

- requirements control
  - **requirements captured in database**
  - **user groups review and prioritize requirements**
  - **RAT - requirements analysis team - involves users**
  - **traceability is an important aspect**
- ICD control
  - **formal control of ICD documents**
  - **wrote up front and closely controlled**
  - **lots of I/Fs**
- CM/QA
  - **extensive database maintained**
  - **tie requiremets to source**
  - **trace to ICD**
- "systems engineering"
  - **team approach to working issues**
  - **technical exchange meetings**
  - **interfaces**
  - **achieve commonality of understanding**
- MIL-STDs
  - **lots of tailoring needed**
- Reviews
  - **extensive government involvement in reviews, wlakthroughs, evaluations**
  - **objectives stated (process, product)**
  - **use of DRR (document review report) database on Mac**
    - ***entered, reviwed, Xref'd, answered, adjudication***
  - **users invited**
  - **database of action items and resolution**
  - **task list database**
  - **rating checklists**
  - **summary observations**
  - **readiness reviews (pre-reviews 1 month ahead)**
- early interface testing
  - **via classified link with contractor**

        - demonstrations
            - **at some sites**
            - **hot test the SW and interfaces**
            - **staged integration**
            - **5 mos before IOC**
        - hardware configuration verification reviews (CVRs)
            - **check revision level of HW, OS, DB for consistency**
            - **understand interrelationships, depndencies in revisions**
            - **manage board level changes to accommodate SW revisions**
    - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
        - writing it down
        - extensive use of databases
        - "we can build complex systems if only we are organized about it"
    - ***What makes you successful, good, valuable to the organization?***
    - ***What are the most difficult aspects of your job? How are they difficult?***

5. **Individual methods**
    - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
        - extensive use of databases
    - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
6. **Individual automation**
    - ***What computers do use in your job? What others are available to you?***
        - portable Mac
    - ***What automated tools do you use today?***
        - Mac tools
            - **Filemaker**
            - **Oracle**
            - **Excel**
            - **Powerpoint**

            - MacDraw
    - *How much of your work involves using the computer?*

7. **Interactions with other individuals, organizations**
    - *Who do you typically work with? What other individuals or roles do you frequently interact?*
        - close working relationship with contractor and users
    - *What interaction do you have with other roles or organizations?*
        - contractor
        - users
    - *What type of interaction is it (voice, documents, etc.)?*
        - reviews, meetings, documents, databases
    - *How much of your time is consumed with this interaction?*
8. **Organization automation and connectivity**
    - *What tools do the other organizations that you work with use? On what computers?*
        - requirements database is reviewed by users
    - *Do you have any electronic connectivity with these systems?*
        - yes
9. **Method improvement**
    - *If you could improve any aspect of your job, what would it be?*
        - more prototyping of user interface earlier
        - have development facility at RL
10. **Automation improvement**
    - *What automated tools or aids do you wish you had?*
        - more automated tools
            - **user interface prototyping**
            - **code analyzers**
        - electronic RFPs
        - better tool integration
            - **interoperability, transfer between tools**
11. **Classified Information**
    - *How much of your job involves classified information?*
        - significant amount
12. **Other comments/observations**

**Interview questions 3.**

1. **Interviewee profile**
   - ***Name, organization, position, educational background, years experience***
     - BS EE, MS EE
     - 6 yrs at RL, previously in industry
     - moved from specialty to systems engineering
2. **Application area, systems**
   - ***What are the most difficults aspect of building these kinds of systems?***
3. **Individual roles and responsibilities**
   - ***Describe what you do. What are your areas of responsibility?***
     - test and integration focus
   - ***How do you fit into the organization?***
   - ***What information do you require or use in your job?***
     - interface documents (5 drawers)
     - test documentation (plans, descriptions, procedures)
     - 2167 documents
     - SDFs
4. **Individual activities**
   - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
     - manage interfaces
       - **coordinate with other systems**
       - **baseline interface documents**
       - **manage with various segments**
         - ***"one no vote stops the game"***
     - program management
       - **weekly meetings of segments**
       - **issues, problems**
       - **coordination**
     - testing
       - **approve contractor test plan, description, procedures**
       - **test readiness review**

        - **documentation review**
        - **witness factory acceptance at contractor facility**
        - **write internal test procedures**
        - **in-house testing**
  - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
    - documentation is the most important
  - ***What makes you successful, good, valuable to the organization?***
    - "bad cop"
  - ***What are the most difficult aspects of your job? How are they difficult?***

5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
   - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
6. **Individual automation**
   - ***What computers do use in your job? What others are available to you?***
   - ***What automated tools do you use today?***
     - E-mail
     - word processors
     - file compare
     - CM tool
   - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
     - contractor
       - **originally had problem**
       - **taught contractor**
       - **"in the contractor's best interest"**
   - ***What interaction do you have with other roles or organizations?***
   - ***What type of interaction is it (voice, documents, etc.)?***

        - FAX, E-mail, meetings, reviews
    - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***
        - "would not do anything differently"
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
11. **Classified Information**
    - ***How much of your job involves classified information?***
12. **Other comments/observations**

        - knows where to look for information, who to talk to

4. **Individual activities**
    - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
        - requirements understanding, decomposition and tracking
            - **80% at the beginning, 20% later**
        - By type of work
            - **20% creative**
                - ***option generation, design, algorithms***
                - ***pictures are very important (e.g., block diagram)***
            - **30% analysis**
                - ***specific analysis to identify goodness***
                - ***finding information***
            - **20% writing, documenting**
            - **30% other**
                - ***coordination***
                - ***planning***
    - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
        - Giving them (the fleet) what they need
    - ***What makes you successful, good, valuable to the organization?***
        - hands-on system development experience makes a good systems engineer
            - **lab experience**
            - **testing**
            - **on-ship deployment situation**
    - ***What are the most difficult aspects of your job? How are they difficult?***
        - requirements tracking (see above)
5. **Individual methods**
    - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
        - basic top-down, decomposition and allocation

# Appendix D. NADC Interviews

**Interview questions 1.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - systems engineer
        - BSEE, MBA
        - 27 years at NADC
        - moved from functional area (acoustics) into systems engineering
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
        - Requirements tracking is the greatest problem
            - **understanding the requirement at the beginning**
            - **requirements decomposition**
            - **requirements change can be very rapid**
                - ***technology (e.g., sensor) change***
                - ***threat change***
            - **traceability must address multi-level security**
3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
        - develop system specification from operational requirements
            - **system specification**
            - **interface definitions**
            - **test plan/req's/specs/procedures**
                - ***"requirements must be written to be tested"***
        - task specific analyses from analysis organizations
            - **define specific questions to be answered**
    - ***How do you fit into the organization?***
        - systems engineer
    - ***What information do you require or use in your job?***
        - full range of life cycle information is needed
        - keeps 5% in head, researches the other 95%

        - knows where to look for information, who to talk to

4. **Individual activities**
    - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
        - requirements understanding, decomposition and tracking
            - **80% at the beginning, 20% later**
        - By type of work
            - **20% creative**
                - *option generation, design, algorithms*
                - *pictures are very important (e.g., block diagram)*
            - **30% analysis**
                - *specific analysis to identify goodness*
                - *finding information*
            - **20% writing, documenting**
            - **30% other**
                - *coordination*
                - *planning*
    - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
        - Giving them (the fleet) what they need
    - ***What makes you successful, good, valuable to the organization?***
        - hands-on system development experience makes a good systems engineer
            - **lab experience**
            - **testing**
            - **on-ship deployment situation**
    - ***What are the most difficult aspects of your job? How are they difficult?***
        - requirements tracking (see above)
5. **Individual methods**
    - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
        - basic top-down, decomposition and allocation

        - analysis (actually performed by analysis personnel or technologists)
        - basic system requirements analysis
            - **understanding**
            - **develop options**
            - **test requirements**
            - **analysis**
        - basic systems engineering skills are transferrable to other application domains
            - **"systems engineers are somewhat flexible"**
    - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***

6. **Individual automation**
    - ***What computers do use in your job? What others are available to you?***
        - rarely use computer because most work involves classified information
    - ***What automated tools do you use today?***
    - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
    - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
        - NAVAIR (customer)
        - supporting analysis and technologists
    - ***What interaction do you have with other roles or organizations?***
        - work agreements formalize specific tasking
    - ***What type of interaction is it (voice, documents, etc.)?***
    - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***

10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
11. **Classified Information**
    - ***How much of your job involves classified information?***
        - 99% is classified
            - **threat info is Secret**
            - **mission, design is Confidential**
        - Classified information is an impediment to automation
            - **Cannot use PCs in office for classified processing**
            - **use hardcopies primarily**
            - **cannot use networking on classified machines**
12. **Other comments/observations**

**Interview questions 2.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - Systems engineer
        - BS ME '58, MS EE '69
        - moved from specialty area (aerial photography) to simulation to project engineerg to systems engineer
        - has worked a single program (LAMPS) since '68
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
        - developing the requirements through implementation and testing
        - begin with the threat (operational requirement) stated in terms not familiar to engineers
        - convert threat into system spec within the constraints of money, schedule, politics
3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***

    - moderator of the multi-functional team (see below)
    - define and prioritize tradeoffs and analyses
    - responsible for the system spec
    - monitor contractor during development
    - contribute to test plans and procedures
    - responsible for integration
  - ***How do you fit into the organization?***
    - systems engineer
  - ***What information do you require or use in your job? What information do you generate?***
    - vast amount of information needed, no limit.
    - system spec

4. **Individual activities**
   - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
     - 2 hrs/day prep for multi-functional team meeting: review, planning
     - 2 hrs/day team meeting to refine timelines
     - 4 hrs/day create minutes, synthesize inputs
     - mentoring
       - **answering questions - 4 dropins, 20 phone calls per day**
   - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
     - overall engineering skills
     - real world experience
     - attitude (eclectic/generalist)
   - ***What makes you successful, good, valuable to the organization?***
     - experience and insight
     - intuition as to the problems
   - ***What are the most difficult aspects of your job? How are they difficult?***
     - constraints
       - **technology**
       - **bureacracy, acquisition process, lead time**

  - systems are getting more complex: you can't keep it all in your head
  - more reliance on the subsystem engineers
  - watching the development being done by others

5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
     - form a multi-functional team (including analysis and technology participants) (10-20 people)
     - develop the system scenario (mission down to minutes)
     - hypothesize a system (people, systems, and interactions)
     - develop "numbers" for the spec
     - develop spec (preformance spec) and system architecture
     - trade studies and analyses are accomplished or spun off from the meeting
       - **models applied by the analysts and technologists (e.g., cost, effectiveness models)**
   - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
     - impart experience, war stories
     - interaction skills
     - specific assignments (to learn, depending on the individual)
     - open-minded interest (the attitude)
6. **Individual automation**
   - ***What computers do use in your job? What others are available to you?***
     - general tools (word processor, spreadsheets, database)
   - ***What automated tools do you use today?***
   - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
   - ***What interaction do you have with other roles or organizations?***
   - ***What type of interaction is it (voice, documents, etc.)?***

- ***How much of your time is consumed with this interaction?***

8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***
        - reduce the time and effort of the collaborative process
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
        - timeline, graphics tools
        - meeting tools
        - tools to shorten the effort to scribe meeting
        - looking for more powerful systems (sun workstations, desktop, CAD/CAM, requirements tools
11. **Classified Information**
    - ***How much of your job involves classified information?***
        - meeting is at secret level
        - mostly unclassified information
12. **Other comments/observations**

**Interview questions 3.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - systems engineer
        - BS,MS Physics
        - 20 yrs experience, stared as subsystem engineer, moved into front-end analysis and then to systems engineering
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
        - change is the case rather than the exception

        - **environment, threat change**
        - **technology change**
        - **requirements is often out of date at the start (old threat)**
    - these systems are getting more complex
        - **ability to control, keep track of everything**
        - **integration**
        - **more players**
        - **competition greater for larger systems**
        - **ripple effect greater for change**
        - **operator is not getting more complex - must try to make the system appear less complex**
    - keeping track of everything is a major thing
        - **change and ripple effect**
        - **requirements, design, criteria, test**
    - making a complex system simple for the operator

3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
        - start with mission analysis and translate into 1st tier engineering requirements
        - develop performance spec
        - overall architecture for the upgrades
        - design allocation, interface specs
        - monitor contractor
        - participate in system V&V with contractor, in the lab
        - do demonstration in lab
            - **test plans and procedures**
            - **simulation**
            - **mathematical simulation**
    - ***How do you fit into the organization?***
        - systems engineer
    - ***What information do you require or use in your job? What information do you generate?***

4. **Individual activities**

- ***What activities consume most of your time? What is the "pie chart" describing what you do?***
  - technical ccordination 10-20%
  - analysis, tradeoffs 70-80% at front end, 10-20% thereafter
  - design and performance specs to finalize configuration items
- ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
  - add credibility in the overall design
  - pragmatic santity check of requirements (on the Gov't side) of design (on the contractor side)
  - glue, bridge between everybody
- ***What makes you successful, good, valuable to the organization?***
- ***What are the most difficult aspects of your job? How are they difficult?***
  - keeping track of requirements into design and FSD
    - **we never have a team big enough to track**
  - getting measurable performance requirements
  - coordination
  - technology assessment
    - **performance, size, weight, power, type of technology, funding**

5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
     - requirements analysis
       - **functional flow**
       - **hierarchical list of functions**
       - **operational sequence**
       - **partitioning and allocation**
       - **block diagrams (using standard symbols)**
       - **timelines**
         - ***used to trace timeline to peformance requirements***
         - ***typically focuses on a on busy period***
     - rely on others to do most of the analysis

- **models used in mission analysis**
  - ***accepted Navy mission models***
  - ***math models***
  - ***local models***
- **some statistical analysis used in risk reduction**

- risk reduction factors tracked throughout the development
  - **10-12 parameters (criteria) tracked throughout the lifecycle (e.g., weight)**

- ***If you were training someone in your job, what would you teach him/her? What advise would you give?***

6. **Individual automation**
   - ***What computers do use in your job? What others are available to you?***
   - ***What automated tools do you use today?***
   - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
   - ***What interaction do you have with other roles or organizations?***
   - ***What type of interaction is it (voice, documents, etc.)?***
   - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
   - ***What tools do the other organizations that you work with use? On what computers?***
   - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
   - ***If you could improve any aspect of your job, what would it be?***
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
      - greatest tool needs at the front end
        - **this time is human interaction intensive**
        - **too much input is needed**
          - ***intuition***
          - ***knowledge***

            - *experience*

11. **Classified Information**
    - ***How much of your job involves classified information?***
        - front end typically classified, development unclassified
        - if tool cannot handle classified information, then it couldn't be used 90% of the time
12. **Other comments/observations**

**Interview questions 4.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - Project engineer
        - BS CS/Math, ongoing MS EE/Computer Design
        - 5 yrs experience (@NADC), 1 yr as project engineer
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
        - deciding on what DIDs and MIL-STDs to include in contracts
            - **deciding on applicability**
            - **no central spot for DIDs and STDs**
3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
        - Write SOWs
        - Develop options paper (in response to operational requirements)
            - **cost, technical options**
        - Evaluate operational requirements
            - **ROM cost estimate**
            - **perform operational analysis**
            - **prepare briefing package and letter**
        - Write and execute contracts
            - **with NAVAIR, contractor, internal organizations**
        - Evaluate contractor performance

- *How do you fit into the organization?*
    - project engineer
- *What information do you require or use in your job? What information do you generate?*

4. **Individual activities**
    - *What activities consume most of your time? What is the "pie chart" describing what you do?*
        - Coordination 90%
            - **40% technical, 50% nontechnical**
            - **45% internal, 45% external**
            - **80% informal, 10% formal**
        - writing SOWs 5%
    - *What do you consider the most important aspects of your job, the things that are most critical to having a successful system?*
        - making NAVAIR happy
            - **right kind of support**
            - **ease their workload**
    - *What makes you successful, good, valuable to the organization?*
    - *What are the most difficult aspects of your job? How are they difficult?*
        - a lot of "clerical" duties (letters, documents)
5. **Individual methods**
    - *What sort of methods or techniques do you use in your job (with respect to the activities described above)?*
    - *If you were training someone in your job, what would you teach him/her? What advise would you give?*
        - go to the experts
            - **if you don't ask you won't go anywhere"**
        - be assertive, ask questions
6. **Individual automation**
    - *What computers do use in your job? What others are available to you?*
    - *What automated tools do you use today?*
        - most used:

        - **Word**
        - **Graphics**
        - **Lotus**
    - Occasional use
        - **COCOMO**
        - **Harvard Project Manager**
        - **Dbase**
- ***How much of your work involves using the computer?***

7. **Interactions with other individuals, organizations**
    - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
        - engineering support organizations (technology, analysis)
        - systems engineer
            - **close working relationship**
            - **"a wealth of information"**
            - **directs technically**
            - **knows programmatics**
            - **draw on his expertise**
            - **knowledge of the process**
            - **specific engineering issues**
        - contractors
    - ***What interaction do you have with other roles or organizations?***
    - ***What type of interaction is it (voice, documents, etc.)?***
        - voice, written
        - Email used for informal communications
            - **problems in using it for formal communications (signatures, legal proof)**
    - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***

10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
        - database for DIDs, CDRLs
11. **Classified Information**
    - ***How much of your job involves classified information?***
        - OR is typically Secret
        - DOPS Secret/Confidential
        - contractor performance 10% classified
        - SOW is sensitive, not classified
12. **Other comments/observations**

**Interview questions 5.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - project engineer
        - BS Physics '59
        - initally I&T, shipboard use
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
        - managing interfaces
            - **only way to manage changes in all of the different programs**
            - **ICWG participation is in everybody's contract**
        - change management may be a future problem
            - **currently handled manually**
            - **changes contribute to imperfect understanding**
3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
        - "what ever people throw at me"
        - single point of contact for the program

        - integrated software product interfaced with two different systems
        - protect NAVAIR - make the program successful
        - interface definition
            - participate in ICWG with all players
    - *How do you fit into the organization?*
        - project engineer, somewhat functioning also as a systems engineer
    - *What information do you require or use in your job? What information do you generate?*
        - generates
            - information in response to ICWG action items
            - IRS
            - SSS

4. Individual activities
    - *What activities consume most of your time? What is the "pie chart" describing what you do?*
        - coordination 30%
        - planning 10-15%
        - budget issues 5%
        - understand and modify tasking priorities 15-20%
        - directing other people 30%
    - *What do you consider the most important aspects of your job, the things that are most critical to having a successful system?*
        - Interface definitions is the leverage
    - *What makes you successful, good, valuable to the organization?*
    - *What are the most difficult aspects of your job? How are they difficult?*
        - maintaining proper priorities
            - priorities may change internally or externally
        - scheduling
            - not enough time ot properly schedule tasks, estimate
            - impact of scheduling

5. **Individual methods**
    - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
        - engineering notebook (chronological)
        - calendar (the calendar is an index into the notebook)
    - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
        - core system engineering perspective
        - fundamentals
6. **Individual automation**
    - ***What computers do use in your job? What others are available to you?***
    - ***What automated tools do you use today?***
    - ***How much of your work involves using the computer?***
        - "not really PC-oriented"
7. **Interactions with other individuals, organizations**
    - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
        - customer (NAVAIR)
        - internal organization
            - **sensor development**
            - **software development & integration**
            - **project engineers**
        - external organization
            - **aircraft integrators**
            - **prime contractor**
            - **Navy computer program office**
        - internal support
            - **senior systems engineers**
            - **junior engineers**
    - ***What interaction do you have with other roles or organizations?***
        - distance seems to matter in communication
            - **leads to imperfect understanding**
            - **corporate cultural differences in interpretation**

- ***What type of interaction is it (voice, documents, etc.)?***
    - voice, documents
    - fax is used a lot
- ***How much of your time is consumed with this interaction?***

8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
        - requirements tracking
        - "keep it simple"
11. **Classified Information**
    - ***How much of your job involves classified information?***
        - vast majority is unclassified, some Confidential, rarely Secret
12. **Other comments/observations**
    - ***computer literacy***
        - probably the worst case scenario for automation

**Interview questions 6.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - project engineer
        - BS Physics
        - test engineer to hardware engineer to systems engineer (5 yr), to project engineer (2 yr)
        - started with small business, moved to a larger contractor and then to NADC -- all dealing with the same basic system

2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
        - hardware engineer performing diagnosis
            - **system level debug, partition to HW/SW and delegate**
                - ***technician for hardware changes***
                - ***computer scientist for software changes***
        - system engineer (software only changes)
            - **generate ECPs**
                - ***poll fleet to understand change need, if necessary***
                - ***generate ECP based upon interactions***
                    - ***varying lengths to several feet of paper***
                    - ***operational requirement - typically classified***
                    - ***addendums for blocks, as needed***
                - ***perform integration assessment***
                    - ***most hardware changes result in software mods***
        - project engineer
            - **"the challenge" "been very difficult"**
                - ***further behind in automation***
                - ***large number of methods, policies***
                - ***6 financial reporting systems to deal with requiring basically the same data***
            - **relate ECP to schedule**
                - ***planning with contingencies***
                - ***manage changes***
                - ***what ifs***
            - **interface to NAVAIR PMA**
            - **review ECPs**
                - ***operational acceptability***
                - ***technical acceptability***
                - ***budget***
    - ***How do you fit into the organization?***

    - project engineer also functioning as a hardware engineer and systems engineer
  - ***What information do you require or use in your job? What information do you generate?***

4. **Individual activities**
   - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
     - 4hr/wk read Email
     - 4hr/wk COTR duties, status reports, etc.
     - 8 hr/wk correspondence, routing, organizing
       - **18" of paper per week**
     - 16 hrs/wk technical advise, reviewing products, acceptance test, time in the lab
     - 8 hr/wk personal capital investment
       - **tools surveys**
       - **training**
       - **build, modify tools**
   - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
   - ***What makes you successful, good, valuable to the organization?***
     - hands-on experience with the system makes the difference
     - in the lab, are the implementers building the right product
     - put reality into the whole thing
   - ***What are the most difficult aspects of your job? How are they difficult?***
5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
   - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
     - use logic analyzer
     - familiarize with computer system to generate management reports
     - boolean logic (computer science at the machine code level)

        - fundamentals as needed

6. **Individual automation**
    - ***What computers do use in your job? What others are available to you?***
        - PC 286
        - avionics lab
    - ***What automated tools do you use today?***
        - Email
        - lotus
    - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
    - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
    - ***What interaction do you have with other roles or organizations?***
    - ***What type of interaction is it (voice, documents, etc.)?***
    - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
        - tools to support and expand roles and capabilities
            - **able to "define the ideal system"**
        - automation is the biggest headache
            - **tools not tailored**
            - **left out critical things (e.g., flowchart generator ignores machine code)**
            - **reverse engineering needed**
        - Hardware tool
            - **CADKey system**

        - *"levels" for various views and ability to overlay the views*
        - *ability to digitize drawings*
    - software tools
        - **generate code from flowcharts**
        - **interactive debug**
        - **generate documentation**
    - systems tools
        - **use software tools to generate a logical model, checkout**
    - integration
        - **future step to relate logical model and physical/ mechanical/ electrical views**

11. **Classified Information**
    - *How much of your job involves classified information?*
        - 5% classified personally
        - supporting people 50% classified
            - **3-4 out 8 people deal with classified information**
        - issues
            - **handling, routing, logging**
            - **finding it amongst several repositories**
12. **Other comments/observations**
    - *Computer lieteracy*
        - probably a best case tool user and tailorer

**Interview questions 7.**

1. **Interviewee profile**
    - *Name, organization, position, educational background, years experience*
        - project engineer
        - BS CS/Aviation, MS engineering
        - 6 years at NADC, started as tester, to task leader, 1 1/2 yrs as project engineer
            - **"do whatever is needed"**

2. **Application area, systems**
   - ***What are the most difficults aspect of building these kinds of systems?***
3. **Individual roles and responsibilities**
   - ***Describe what you do. What are your areas of responsibility?***
     - scheduling
     - presentations
     - technical direction to lower level individuals
       - **do research**
       - **plans, procedures**
       - **tech memos**
       - **write software**
     - tasking
       - **NAVAIR gives task**
         - ***varying length tasks 1 week to 2+ years***
       - **work with branch heads to identify resources**
       - **writeup task assignments (deliverables, cost, schedule)**
       - **coordinate and monitor the task effort**
   - ***How do you fit into the organization?***
     - project engineer
   - ***What information do you require or use in your job? What information do you generate?***
     - lots of info is needed
     - key is how to find it
       - **ask others**
       - **systems engineer**
4. **Individual activities**
   - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
     - scheduling 50-75%
     - technical direction <25%
       - **others are always asking for guidance**
   - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***

- ***What makes you successful, good, valuable to the organization?***
    - a good leader
    - not afraid to work
    - do what it takes to get the job done
    - step up to more responsibility
    - her aviation background is the basis for understanding the application
        - **at a disadvantage because of lack of deep application knowledge**
- ***What are the most difficult aspects of your job? How are they difficult?***
    - not having enough time
        - **often meetings are a distraction**
        - **1/2 of tool use is nonproductive**
    - time management and organization (filing)

5. **Individual methods**
    - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
        - process has been relatively stable
    - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
        - teach day-to-day operations
        - keep informed, up-to-date
6. **Individual automation**
    - ***What computers do use in your job? What others are available to you?***
    - ***What automated tools do you use today?***
        - Email used every day
    - ***How much of your work involves using the computer?***
        - frequent computer user
            - **Email**
            - **scheduling**

7. **Interactions with other individuals, organizations**
    - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
        - very close relationship with the systems engineer
            - **systems engineer is the technical lead**
            - **keep up mutual status**
            - **team cooperation**
    - ***What interaction do you have with other roles or organizations?***
        - 50% internal interaction
        - 50% external interaction
    - ***What type of interaction is it (voice, documents, etc.)?***
        - documents, papers, voice
        - Email
            - **setup meetings with sponsors**
            - **status**
            - **questions and answers**
            - **easy way to route information**
            - **review of working papers**
            - **trip reports**
            - **progress reports**
    - ***How much of your time is consumed with this interaction?***
8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
        - good project scheduler
            - **tools must be user friendly**
                - ***ease of use is more important than a wide variety of capabilities***

11. **Classified Information**

- ***How much of your job involves classified information?***

12. **Other comments/observations**

## Appendix E. IBM Interviews

**Interview questions 1.**

1. **Interviewee profile**
   - ***Name, organization, position, educational background, years experience***
     - PhD Advanced Technology/System Science
     - 8 yrs BM experience
2. **Application area, systems**
   - ***What are the most difficults aspect of building these kinds of systems?***
     - interdisciplinary communication
     - change management
       - **communicate changes**
       - **flow down changes from the team**
       - **maintain traceability**
     - vendors (perhaps our biggest problem)
       - **giving vendors a good requirements specification**
       - **communications**
         - ***formal means restricted communications***
         - ***no informal communications***
3. **Individual roles and responsibilities**
   - ***Describe what you do. What are your areas of responsibility?***
     - performs advanced study contracts
     - proposals
     - systems engineer at front-end of system developments
   - ***How do you fit into the organization?***
     - Avionics Systems organization
     - systems engineering reports from a matrix organization to the program manager
   - ***What information do you require or use in your job?***
     - Operational Need Document
     - Proposal
     - Work Breakdown Structure

- Technical Management Plan (equivalent to SEMP)
- New Program Startup Guide
- System Specification
- Segement Specifications
- Design Analysis Report
- System Design Document
- previous proposals and documents

4. **Individual activities**
   - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
     - Pre-RFP
       - **talking to the customer**
       - **refine operational need**
       - **do mission analysis**
       - **perform market research**
     - RFP-Proposal
       - **appoint a lead engineer**
       - **put requirements into requirements management system**
       - **allocate each requirement to "owners"**
       - **clarify the requirements**
       - **identify technical performance measures**
       - **identify and rate (L,M,H) risks**
       - **do Technical Management Plan**
       - **establish WBS**
       - **setup baseline control procedures**
       - **form and chair Engineering Integration Working Group**
       - **do Systems Spec, Segment Specs, mission analysis, maybe early trade studies, some System Design Document**
     - Program start-up
       - **conduct proposal requirements review**
         - *did any requirements change*
       - **bring together team**
         - *often different from proposal*
         - *hopefully the same lead*

            - *communicate requirements, plans and problems*
    - Program execution
        - **startup program according to New Program Startup Guide**
        - **"fighting fires"**
            - ***trying to get users, customer and IBM to agree on requirements clarifications***
- ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
- ***What makes you successful, good, valuable to the organization?***
- ***What are the most difficult aspects of your job? How are they difficult?***
    - communication
    - change management

5. **Individual methods**
    - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
        - reuse of proposals, specifications and other information
        - multi-view requirements analysis
            - **functional - PSA templates**
            - **physical - system block diagrams**
            - **operational - homegrown methods**
            - **three views manually associated**
        - requirements allocation and "ownership"
    - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
6. **Individual automation**
    - ***What computers do use in your job? What others are available to you?***
    - ***What automated tools do you use today?***
        - requirements management system (internal)
            - **relational database manager**
            - **automatically generates requirements spec**
        - "se help" - generic plans for use on programs
        - PSA

    - CAD for system block diagrams
    - APL tools
    - decision pad
    - virtual avionics prototyping
    - ABE - Teknowledge queuing model
    - RESQUE queing model
  - ***How much of your work involves using the computer?***

7. **Interactions with other individuals, organizations**
  - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
    - other systems engineers
    - customer
    - program manager
    - specialty engineers
    - vendors
  - ***What interaction do you have with other roles or organizations?***
    - formal communications only with vendors
    - Engineering Integration Working Group
      - **chaired by systems engineering**
      - **members consist of leads for each discipline**
      - **meet weekly**
      - **negotiation is primary activity**
    - Program Office Directives
      - **formal communication to team**
      - **e.g., change in requirements**
    - Systems Engineering Directives
      - **formal communication to team**
      - **technical change or clarification**
    - informal team communications
      - **communicate requirements, plans and problems**
  - ***What type of interaction is it (voice, documents, etc.)?***
    - meetings - informal communication
    - specs and directives - formal communication
  - ***How much of your time is consumed with this interaction?***

8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
11. **Classified Information**
    - ***How much of your job involves classified information?***
12. **Other comments/observations**

**Interview questions 2.**

1. **Interviewee profile**
    - ***Name, organization, position, educational background, years experience***
        - BS EE, MS CE
        - 10+ years experience at IBM
2. **Application area, systems**
    - ***What are the most difficults aspect of building these kinds of systems?***
        - interfacing with people
            - **differences**
            - **interfaces**
            - **information**
            - **schedule, resources**
            - **traceability**
            - **HW availability**
        - "keeping it all together"
            - **>1000 requirements**
            - **derived requirements**
            - **test**
            - **design**

        - **use of requirements, test, etc. databases**

3. **Individual roles and responsibilities**
    - ***Describe what you do. What are your areas of responsibility?***
        - avionics systems engineer
    - ***How do you fit into the organization?***
        - part of systems engineering team
    - ***What information do you require or use in your job?***
        - mission requirements
        - system requirements
        - design description document
            - **includes rational, descriptive information**
        - integration and test plan
        - interface control documents
4. **Individual activities**
    - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
        - understand operational need
        - define high level system requirements
        - allocate requirements to subsystems
        - do lower level requirements specification
        - write design description document
        - do interface control documents
        - work with subsystem designer during subsystem design
        - write integration and test plans and procedures
        - activities:
            - **60% interactions**
            - **20% meetings**
            - **20% doing real productive work (i.e., doing work products)**
    - ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
        - defining the requirements
            - **"...could always do that better"**
            - **derived requirements drift away at the end because they are not required to be tested**

- ***What makes you successful, good, valuable to the organization?***
  - requirements definition
  - maintaining baseline
- ***What are the most difficult aspects of your job? How are they difficult?***

5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
     - multi-view requirements analysis
       - **functional - SSS view**
       - **physical -HW diagrams, design description document**
       - **operational - timeline**
       - **three views manually associated**
   - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
     - mentoring
     - training on systems engineering process and methodology
6. **Individual automation**
   - ***What computers do use in your job? What others are available to you?***
     - PC AT personally, PS/2 for team
   - ***What automated tools do you use today?***
     - editor
     - PROFS (e-mail, file transfer, calendar, meeting mgmt)
     - Bookmaster - documentation, charts
     - Markup tool
     - PSL/PSA
     - Drawmaster graphics tool - block diagrams, flowcharts
     - PERT
     - requirements management system
   - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***

- systems engineering team
  - **12-15 systems engineers**
  - **10+ programmers**
  - **6+ hardware engineers**
  - **quality, reliability, logistics, training**
- subsystem designer
- IV&V organization
- software engineer
- HW lab/integrators
- test engineers
- speciality disciplines (quality, reliability, logistics)

- ***What interaction do you have with other roles or organizations?***
  - talking
  - meetings
  - reviews
  - E-mail
  - electronic markup of documents
- ***What type of interaction is it (voice, documents, etc.)?***
  - talking, meetings, reviews, E-mail
- ***How much of your time is consumed with this interaction?***
  - 60% interactions
  - 20% meetings

8. **Organization automation and connectivity**
   - ***What tools do the other organizations that you work with use? On what computers?***
   - ***Do you have any electronic connectivity with these systems?***
     - yes, facility network
9. **Method improvement**
   - ***If you could improve any aspect of your job, what would it be?***
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
11. **Classified Information**
    - ***How much of your job involves classified information?***
12. **Other comments/observations**

**Interview questions 3.**

1. **Interviewee profile**
   - ***Name, organization, position, educational background, years experience***
     - BS EE
     - 8 yrs at IBM
2. **Application area, systems**
   - ***What are the most difficults aspect of building these kinds of systems?***
     - assessing compliance to requirements
       - **... that it meets an operational solution**
       - **need to first understand the need**
       - **PDR, CDR that designmeets requirements**
       - **simulation has been a help**
       - **keep customer involved throughout**
     - cost and schedule constraints
       - **often have rotating shifts at the end**
3. **Individual roles and responsibilities**
   - ***Describe what you do. What are your areas of responsibility?***
     - "full life cycle" engineer
     - COG engineer - subsystem specs that will be cantracted out
     - integration and test
   - ***How do you fit into the organization?***
     - Avionics systems
   - ***What information do you require or use in your job?***
     - mission requirements
     - PIDS requirements
     - SRS
     - master test plan
     - design analysis report
       - **algorithms**
       - **moding**
       - **higher level functional design**
     - interface requiremets documents

- test procedures

4. **Individual activities**
   - ***What activities consume most of your time? What is the "pie chart" describing what you do?***
     - as COG engineer
       - **develop hardware specs**
       - **contract out**
     - as life cycle engineer
       - **develop requirements - SRS**
       - **integration and test**
         - ***develop test procedures***
         - ***organize the library***
         - ***conduct test to PIDS requirements***
       - **system test**
       - **flight test**
     - integration and test engineer
       - **lead I&T is identified**
         - ***"jack of all trades" at the beginning***
       - **generate master test plan**
         - ***lab configuration***
         - ***general test process, types of test***
         - ***test matrix***
       - **assist lab development folks**
         - ***how it is wired***
         - ***ICDs***
       - **define system test procedures**
         - ***organize by major functions***
         - ***for each requirement in the database***
         - ***generate scenarios for simulators***
       - **review/refine with customer**
       - **do test procedures in laboratory demonstration**
         - ***scenario-based***
       - **document test reports to customer**

- ***What do you consider the most important aspects of your job, the things that are most critical to having a successful system?***
- ***What makes you successful, good, valuable to the organization?***
  - experience in the "gamit"
    - **requirements through test**
    - **keeping design real, rational**
  - work with people
    - **communicate, interface**
  - customer interface
- ***What are the most difficult aspects of your job? How are they difficult?***

5. **Individual methods**
   - ***What sort of methods or techniques do you use in your job (with respect to the activities described above)?***
     - alllocation, traceability
     - simulation
   - ***If you were training someone in your job, what would you teach him/her? What advise would you give?***
6. **Individual automation**
   - ***What computers do use in your job? What others are available to you?***
   - ***What automated tools do you use today?***
     - requirements database for PIDS requirements
     - RMAT - scans and parses out wills and shalls
     - simulators
   - ***How much of your work involves using the computer?***
7. **Interactions with other individuals, organizations**
   - ***Who do you typically work with? What other individuals or roles do you frequently interact?***
     - customer
     - other systems engineers
     - test engineer
   - ***What interaction do you have with other roles or organizations?***
     - can talk with customer directly

        - subcontract out hardware
    - ***What type of interaction is it (voice, documents, etc.)?***
        - personal, meetings
    - ***How much of your time is consumed with this interaction?***
        - "a lot of the job is communication"

8. **Organization automation and connectivity**
    - ***What tools do the other organizations that you work with use? On what computers?***
    - ***Do you have any electronic connectivity with these systems?***
9. **Method improvement**
    - ***If you could improve any aspect of your job, what would it be?***
        - bring team together - cradle to grave
            - **matrix doesn't work**
        - perhaps improve avionics simulator
10. **Automation improvement**
    - ***What automated tools or aids do you wish you had?***
        - better centralized database
            - **compatibility of PC and VM (global databases)**
        - tool for generating boilerplate for test procedures
        - vendor link to requirements database
11. **Classified Information**
    - ***How much of your job involves classified information?***
12. **Other comments/observations**

## References

[AFS89] Air Force Studies Board. *Adapting Software Development Policies to Modern Technology*. Washington: National Academy Press, 1989.

[BLA90] Black, Harlan (editor), et al. TTCP Requirements Engineering and Rapid Prototyping Workshop Proceedings. Hosted by the U.S. Army Communications-Electronics Command (CECOM) Center for Software Engineering. New Jersey: November 14–16, 1989. May, 1990.

[BOE84] Boehm, Barry W, et al. "A Software Development Environment for Improving Productivity." *IEEE Computer*. June 1984. 30–44.

[BOE88] Boehm, Barry W. "A Spiral Model of Software Development and Enhancement." *IEEE Computer*. May 1988. 61–72.

[BOU79] Bouchard, Thomas, J., "Field Research Methods: Interviewing, Questionaires, Participant Observation, Systematic Observation, Unobtrusive Measures," *Handbook of Industrial and Organizational Psychology*, Marvin D. Dunnette (ed), Rand McNally College Publishing Company, Chicago, IL, 1976. 363-413.

[BRO87] Brooks, Frederick P., Jr. "No Silver Bullet: Essence and Accidents of Software Engineering." *IEEE Computer*, Volume 20, Number 4, April 1987. 10–19.

[COD84] Council of Defense and Space Industry Associations (CODSIA). *DoD Management of Mission-Critical Computer Resources*. CODSIA Report 13–82 (Volume I, Task Group Report) to Under Secretary of Defense, Research and Engineering. March 1984.

[CUR87a] Cureton, Bill. "The Future of UNIX As A Platform for CASE." *First International Workshop of Computer-Aided Software Engineering*. Volume 1. Cambridge, Massachusetts. May 27-29, 1987. 211-215.

[CUR87b] Curtis, Bill. "Introduction to Empirical Research on the Design Process in MCC's Software Technology Program." *Empirical Studies of the Design Process: Papers for the Second Workshop on Empirical Studies of Programmers*, MCC Technical Report Number STP-260-87, Sept. 24, 1987. 1-4.

[DEU90] Deutsch, Michael S. "An Exploratory Analysis Relating the Software Project Management Process to Project Success." Hughes Aircraft Company, 1990.

[GAO79] United States General Accounting Office (GAO). *Contracting for Computer Software Development—Serious Problems Require Management Attention to Avoid Wasting Additional Millions*. Report to the Congress by the Comptroller General. FGMSD–80–4. November 9, 1979.

[KRA87] Krasner, Herb, et. al. "Communication Breakdowns and Boundary Spanning Activities on Large Programming Projects." *Empirical Studies of the Design Process: Papers for the Second Workshop on Empirical Studies of Programmers.* MCC Technical Report Number STP-260-87, Sept ember 24, 1987. 46-66.

Also published in *Empirical Studies of Programmers: Second Workshop,* G. Olson, S. Sheppard, and E.Soloway, eds., Norwood, N.J.: Ablex Publishing Corporation, 1987.

[KRA88] Krasner, Herb, et. al. "A Field Study of the Software Design Process for Large Systems," *Communications of the ACM,* Vol. 31, No. 11, November 1988.

[RED84] Redwine, Samuel T., et al. *DoD Related Software Technology Requirements, Practices, and Prospects for the Future.* Institute for Defense Analysis. Prepared for the Office of the Under Secretary of Defense for Research and Engineering. IDA Paper P–1788. June 1984.

[REP89] *Bugs in the Program: Problems in Federal Government Computer Software Development and Regulation,* Staff Study by the Subcommitte on Investigations and Oversight transmitted to the Committe on Science, Space and Technology, U.S. House of Representatives, September 1989.

[SIN90] Singh, Kamar J. *DARPA Initiative in Concurrent Engineering (DICE) - Phase 1.* GE Aircraft Engines. Issued by DARPA/CMO under Contract No. MDA972-88-C-0047. February 9, 1990.

[SWA88] Swanson, E. Burton and Beath, Cynthia M., "The Use of Case Study Data in Software Management Research," *Journal of Systems and Software,* Vol. 8, 1988. 63-71.

[WEI75] Weiss, David M. *The MUDD Report: A Case Study of Navy Software Development Practices.* Naval Research Laboratory (NRL). NRL Report 7909. May 21, 1975.

[WIN88] Winner, Robert I., et al. *The Role of Concurrent Engineering in Weapon Systems Acquisition.* Institute for Defense Analysis. Prepared for the Assistant Secretary of Defense for Production and Logistics. IDA report R–338. December 1988.

☆U.S. GOVERNMENT PRINTING OFFICE 1991-710-033-60213